

Jacobson

DTIC / TR-86 / 5

A

**Proceedings of the
Second Conference on
Computer Interfaces and Intermediaries
for Information Retrieval**

May 28-31, 1986
Boston, Massachusetts

Sponsored by the
Defense Technical Information Center
and the
Massachusetts Institute of Technology

ADA 174000

20071001025

DTIC

Defense
Technical
Information
Center

Office of Information Systems and Technology

Cameron Station, Alexandria, VA 22304-6145

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified/Unlimited			1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY			3. DISTRIBUTION / AVAILABILITY OF REPORT Public Release, Unlimited Distribution	
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE				
4. PERFORMING ORGANIZATION REPORT NUMBER(S) DTIC/TR-86/5			5. MONITORING ORGANIZATION REPORT NUMBER(S)	
6a. NAME OF PERFORMING ORGANIZATION Defense Technical Information Center		6b. OFFICE SYMBOL (If applicable) DTIC-E		7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) Cameron Station Alexandria, VA 22304-6145			7b. ADDRESS (City, State, and ZIP Code)	
8a. NAME OF FUNDING / SPONSORING ORGANIZATION		8b. OFFICE SYMBOL (If applicable)		9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)			10. SOURCE OF FUNDING NUMBERS	
			PROGRAM ELEMENT NO.	PROJECT NO.
			TASK NO.	WORK UNIT ACCESSION NO.
11. TITLE (Include Security Classification) Proceedings of the Second Conference on Computer Interfaces and Intermediaries for Information Retrieval. May 28-31, 1986, Boston, Massachusetts				
12. PERSONAL AUTHOR(S) Carol E. Jacobson, Compiler Shirley A. Witges, Compiler				
13a. TYPE OF REPORT Summary		13b. TIME COVERED FROM TO		14. DATE OF REPORT (Year, Month, Day) 8605
15. PAGE COUNT				
16. SUPPLEMENTARY NOTATION Cosponsored by the Massachusetts Institute of Technology.				
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	SUB-GROUP	*Information Retrieval,*Computer Applications,*Man Computer Interface, Artificial Intelligence,Natural Language, Information Systems, On Line Systems, Gateway Retrieval Systems Interface Design, Expert Systems	
05	02			
09	02			
19. ABSTRACT (Continue on reverse if necessary and identify by block number) These proceedings consist of papers and product descriptions presented at the "Second Conference on Computer Interfaces and Intermediaries for Information Retrieval," 28-31 May 1986, at the Boston Park Plaza Hotel and Towers, Boston, MA. The Conference was jointly sponsored by the Defense Technical Information Center and the Massachusetts Institute of Technology. The purpose of this conference was to bring together experts in the field of user interfaces and gateways for information retrieval systems. Twenty eight speakers presented papers in the following areas of interest: artificial intelligence in retrieval, gateways, interfaces, common command language, and natural language.				
20. DISTRIBUTION / AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input checked="" type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION Unclassified/Unlimited	
22a. NAME OF RESPONSIBLE INDIVIDUAL Carol Jacobson			22b. TELEPHONE (Include Area Code) (202)274-7661	22c. OFFICE SYMBOL DTIC-E

SECURITY CLASSIFICATION OF THIS PAGE

SECURITY CLASSIFICATION OF THIS PAGE

Proceedings of the
SECOND CONFERENCE ON
COMPUTER INTERFACES AND INTERMEDIARIES FOR
INFORMATION RETRIEVAL

The Boston Park Plaza Hotel and Towers,
Boston, Massachusetts

28-31 May 1986

Sponsored by the
Defense Technical Information Center
and the
Massachusetts Institute of Technology

TABLE OF CONTENTS

Conference Agenda	<u>Page</u> 1
CONFERENCE INTRODUCTION	
MARTHA WILLIAMS, Keynote Speech, "Evaluation of Front-Ends and Gateways"	365
CAROL FENICHEL, "Intermediary Information Systems: The Range of Options From the User's Point of View"	9
SESSION I, AI in Retrieval	
GIO WIEDERHOLD, "Structural vs. Application Knowledge for Improved Database Interfaces"	17
NICK BELKIN, "What Does it Mean for an Information System Interface to be Intelligent?"	97
LINDA SMITH, "Machine Intelligence vs. Machine-Aided Intelligence as a Basis for Interface Design"	107
MICHAEL PINCUS/KATHY PINCUS, "Functionally Intelligent Interface: A Comprehensive Model of Human Cognition Incorporating Both Possibilistic and Probabilistic Functioning"	113
BRUCE CROFT/ROGER THOMPSON, "An Overview of the Intelligent Interface for Information Retrieval (IIIR) Document Retrieval System"	123
EDWARD FOX, "A Design for Intelligent Retrieval: The CODER System"	135
LIONEL BERNSTEIN, "Natural Language Querying of a Medical Knowledge Base"	155

SESSION II, Gateways

VIKTOR HAMPEL , "Gateway Technology"	<u>Page</u> 365
GLADYS COTTER , "The DoD Gateway Information System: Prototype Experience"	173
JUDY HUSHON , "The Evolution of Gateway Technology".	181

VENDOR PRESENTATIONS

Artificial Intelligence Corporation (INTELLECT)	203
Chemical Abstracts Service (CAS ONLINE)	204
Dialog Information Services, Inc. (DIALOGLINK and DIALOG Business Connection)	205
Disclosure Information Group, Inc. (MicroDisclosure).	206
H. W. Wilson Company (WILSEARCH)	207
Institute for Scientific Information (Sci-Mate)	208
KNM, Inc. (SIRE)	209
Telebase Systems Inc. (EasyNet)	210
Minicomputer Systems Incorporated (IRVING Library Network).	211

SESSION III, Interfaces I

RICHARD MARCUS , "Issues for Expert Systems for Retrieval Assistance"	213
CHARLES MEADOW , "OAK -- A New Approach to User Search Assistance"	215
DAVID TOLIVER , "Whether and Whither Micro-Based Front-Ends"	225
RITA BERGMAN , "Dealing with Heterogeneity: An Architecture for Uniform Access/Integrated Presentation".	235

SESSION III, Interfaces II

	<u>Page</u>
BILL MISCHO , "End-User Searching in an Online Catalog Environment"	241
MICHAEL MONAHAN , "The Impact of Networking Standards on Library Systems"	263
DONALD HAWKINS/LOUISE LEVY , "Introduction of Online Searching to End Users at AT&T Bell Laboratories"	275
MARCIA BATES , "Terminological Assistance for the Online Subject Searcher"	285

SESSION IV, Common Command Language

HILARY BURTON , "Developing System Interfaces for an Intelligent Gateway to a Heterogeneous Resource Environment"	295
DINEH MOGH DAM DAVIS , "Issues Raised by Common Command Language Standards"	303
EMILY FAYEN , "The Common Command Language"	311

SESSION V, Natural Language I

GABRIEL JAKOBSON , "Toward Expert Database Front-Ends"	317
RALPH WEISCHEDEL , "Thoughts on the Use of Natural Language Interfaces with Examples From the IRUS/Janus Project"	319
SHARON SALVETER , "A Natural Language Front-End for Database Update"	329

SESSION V, Natural Language II

CRAIG THOMPSON/STEVE MARTIN , "Using Menu-Based Natural Language to Query An Integrated Database Management and Information Retrieval System"	337
GERARD SALTON , "Natural Language Processing in Information Retrieval -- An Outline"	355

	<u>Page</u>
Titles of Papers Presented at the Meeting For Which the Text Does Not Appear in the Proceedings.	365
Author Index.	367
Subject Index	369
Acknowledgements.	371

**THE SECOND CONFERENCE ON
COMPUTER INTERFACES AND INTERMEDIARIES FOR
INFORMATION RETRIEVAL**

The Boston Park Plaza Hotel & Towers, Boston, Massachusetts
28-31 May 1986

Objective and Scope

The purpose of this Conference is to bring together experts in the field of user interfaces and gateways for information retrieval systems. Whereas information retrieval systems offer access to information in diverse heterogeneous databases, access and search of these databases requires a wide variety of protocols and command languages. This is difficult, not only for the trained searcher, but also for the end-user who might like to access the information in databases but who has little time to keep abreast of the latest enhancements to search systems.

One of the most rapidly expanding areas of online information retrieval is the development of software specifically designed to simplify the search process, lower the search costs, save time, reformat retrieved information, and improve search results. Because this software is so new, there is no agreed-upon terminology. It may be referred to as "front-end," "intermediary," "gateway," "user interface," "transparent system," "user-friendly software," or other similar terms. The goal, however, is to assist the trained searcher or enduser with database selection, search formulation, logging on, uploading, downloading, post processing, and record keeping.

Conference speakers will present papers and moderate discussions in areas of interest such as artificial intelligence in retrieval, gateways, interfaces, common command language, and natural language. In addition, vendors will offer presentations concerning specific company products. These will be accompanied by appropriate product demonstrations.

On Friday afternoon, 30 May, the program schedule includes a visit to the Massachusetts Institute of Technology facilities for on-site discussions and limited lab visits.

PRINCIPAL SPEAKERS

Mr. Richard D. Douglas, Director
Office of Information Systems and Technology
Defense Technical Information Center

Dr. Martha Williams, Professor of Information Science,
Coordinated Science Laboratory
University of Illinois at Urbana-Champaign

Dr. Carol Fenichel, Director of the Library
Hahnemann University

AGENDA

WEDNESDAY, 28 MAY

1700-1900 **Registration** -- Boston Park Plaza Hotel & Towers

THURSDAY, 29 MAY

0700 Registration
0750 Conference Convenes -- Arlington/Berkeley Rooms
 Introduction -- **Marjorie Powell**, Program Analyst,
 Defense Technical Information Center
0800 Welcoming Remarks -- **Richard D. Douglas**,
 Director, Office of Information Systems and
 Technology, DTIC
0815 Keynote -- **Martha Williams**, Professor of
 Information Science, Coordinated Science Laboratory,
 University of Illinois at Champaign-Urbana
0845 Overview -- **Carol Fenichel**, Director of the
 Library, Hahnemann University, "Intermediary
 Information Systems: The Range of Options From the
 User's Point of View"

Break

SESSION I AI in Retrieval

0930-1315
Moderator: **Carol Jacobson**, Technical Information Specialist,
 Defense Technical Information Center

Speakers: **GIO WIEDERHOLD**, Professor, Computer Science
 Department, Stanford University, "Structural vs.
 Application Knowledge for Improved Database
 Interfaces"

NICK BELKIN, Professor, School of Communication,
 Information and Library Studies, Rutgers University,
 "What Does it Mean for an Information System
 Interface to be Intelligent?"

LINDA SMITH, Associate Professor, Graduate School
 of Library and Information Science, University of
 Illinois, "Machine Intelligence vs. Machine-Aided
 Intelligence as a Basis for Interface Design"

MICHAEL PINCUS, Chairman of the Board,
 Thunderstone/EPI, Inc. Copresenter, **KATHY PINCUS**,
 President, Thunderstone/EPI, Inc. "Functionally
 Intelligent Interface: A Comprehensive Model of
 Human Cognition Incorporating Both Possibilistic and
 Probabilistic Functioning"

SESSION I, AI in Retrieval. . .continued

Speakers: **BRUCE CROFT**, Associate Professor of Computer and Information Science, University of Massachusetts. Presenter, **Roger Thompson**, Research Assistant, University of Massachusetts. "An Overview of the Intelligent Interface for Information Retrieval (IIIR) Document Retrieval System"

EDWARD A. FOX, Assistant Professor, Department of Computer Science, Virginia Polytechnic and State University, "A Design for Intelligent Retrieval: The CODER System"

LIONEL BERNSTEIN, Professor of Medicine, Center for Educational Development, University of Illinois at Chicago, "Natural Language Querying of a Medical Knowledge Base"

Questions/Answers

1315 **Lunch** (on your own)

SESSION II Gateways

1445-1630

Moderator: **Bill Saunders**, Technical Information Specialist, Defense Technical Information Center

VIKTOR HAMPEL, Project Leader, Technology Information System, Lawrence Livermore National Laboratory, "Gateway Technology"

GLADYS COTTER, Chief, Information Systems Division, Defense Technical Information Center, "The DoD Gateway Information System: Prototype Experience"

JUDY HUSHON, Manager, Washington Office, Research Systems Department, BBN Laboratories Inc., "The Evolution of Gateway Technology"

Questions/Answers

1630

Break

VENDOR PRESENTATIONS

1645-1900 Artificial Intelligence Corporation (INTELLECT)
Chemical Abstracts Service (CAS ONLINE)
Dialog Information Services, Inc. (DIALOGLINK and
DIALOG Business Connection)
Disclosure Information Group, Inc. (MicroDisclosure)
H. W. Wilson Company (WILSEARCH)
Institute for Scientific Information (Sci-Mate)
KNM, Inc. (SIRE)
Telebase Systems Inc. (EasyNet)
Questions and Answers

1900-2030 **Reception/Vendor Demonstrations --** Stanbro Room
(Refreshments will be served)

FRIDAY, 30 MAY
0700 Registration

SESSION III

Interfaces I

0800-1015 Arlington/Berkeley Rooms

Moderator: **Shirley Witges.** Technical Librarian, Defense
Technical Information Center

Speakers: **RICHARD MARCUS**, Principal Research Scientist,
Laboratory for Information and Decision Systems,
Massachusetts Institute of Technology, "Issues for
Expert Systems for Retrieval Assistance"

CHARLES MEADOW. Professor of Library and Information Science, University of Toronto, "OAK -- A New Approach to User Search Assistance"

DAVID TOLIVER, Manager, Development Department,
Institute for Scientific Information, "Whether and
Whither Micro-Based Front-Ends"

RITA BERGMAN, Branch Manager, Research and Systems Business Development, Computer Corporation of America, "Dealing with Heterogeneity: An Architecture for Uniform Access/Integrated Presentation"

Questions/Answers

1015 Break

Interfaces II

1030-1245

Speakers:

BILL MISCHO, Engineering Librarian, Engineering Library, University of Illinois at Urbana-Champaign, "End-User Searching in an Online Catalog Environment"

MICHAEL MONAHAN, Product Manager, Library Systems, GEAC Computers International Inc., "The Impact of Networking Standards on Library Systems"

DONALD HAWKINS, Senior Information Technology Scientist, AT&T Bell Laboratories. Copresenter, **LOUISE LEVY**, Market/Product Development Specialist, Library Network, AT&T Bell Laboratories. "Introduction of Online Searching to End Users at AT&T Bell Laboratories"

MARCIA BATES, Associate Professor, Graduate School of Library and Information Science, University of California at Los Angeles, "Terminological Assistance for the Online Subject Searcher"

Questions/Answers

1245

Lunch -- Stanbro Room

SESSION IV Common Command Language

1415-1545

Moderator:

Allan Kuhn, Program Analyst,
Defense Technical Information Center

Speakers:

HILARY BURTON, Technical Information Specialist, Lawrence Livermore National Laboratory, "Developing System Interfaces for an Intelligent Gateway to a Heterogeneous Resource Environment"

DINEH MOGH DAM DAVIS, Assistant Professor, Computer Information Systems Department, Bentley College, "Issues Raised by Common Command Language Standards"

EMILY FAYEN, Assistant Director for Library Systems, Van Pelt Library, University of Pennsylvania, "The Common Command Language"

Questions/Answers

1545 **Break**

1600 **Bus Transportation to MIT**

1630 **MIT Visit/Reception/Discussion**

1900 Buses return to Boston Park Plaza & Towers
(about)

SATURDAY, 31 MAY

SESSION V
Natural Language I

0830-1015 Arlington/Berkeley Rooms

Moderator: **Gladys Cotter**, Chief, Information Systems
Division, Defense Technical Information Center

Speakers: **GABRIEL JAKOBSON**, Senior Member of Technical
Staff, Computer Science Laboratory, GTE Laboratories,
Inc., "Toward Expert Database Front-Ends"

RALPH WEISCHEDEL, Senior Scientist, Computer
Science Division, BBN Laboratories Inc.,
"Thoughts on the Use of Natural Language Interfaces
With Examples From the IRUS/Janus Project"

SHARON SALVETER, Professor of Computer Science,
Computer Science Department, Boston University, "A
Natural Language Front-End for Database Update"

1015-1030 **Break**

Natural Language II

1045-1145 **CRAIG THOMPSON**, Technical Staff Member, Central
Research Lab, Texas Instruments. Presenter, **STEVE**
MARTIN. "Using Menu-Based Natural Language to
Query an Integrated Database Management and
Information Retrieval System"

GERARD SALTON, Professor, Department of Computer
Science, Cornell University, "Natural Language Query
Processing in Information Retrieval -- An Outline"

Questions/Answers

1145-1230

Wrap-Up
Linda Smith
Richard Marcus
Gerard Salton

Closing Remarks

Richard Douglas, Director, Office of Information
Systems and Technology, Defense Technical Information
Center

Adjourn

INTERMEDIARY INFORMATION SYSTEMS
THE RANGE OF OPTIONS FROM THE USER'S POINT OF VIEW

Carol Hansen Fenichel

From the perspective of an intermediary trying to use commercially-available software effectively, the online environment has become extremely complex. It is difficult to make informed decisions regarding the most appropriate software for a particular situation. While so-called front-end and gateway software has improved greatly in the past two years, the advantages must outweigh the cost and the very real effort involved in learning to use the software, particularly since this software competes with general communications software, "user-friendly" systems, and, in some environments, CD-ROM-based systems.

One important role for information professionals is advisor to end-users. Because they are apt to use only one retrieval system, end-users are the major audience for the stand-alone intermediary programs. The ease with which the programs facilitate getting started (easy logon) is surprisingly important. Software which helps control costs has proven popular with novice users.

For professional searcher intermediaries, the primary advantages of intermediary information systems are the cost-saving features (uploading of strategies) and the post-processing capabilities.

Intermediary Information systems, Gateways, Front-End, Search Intermediaries, End-Users.

1. INTRODUCTION

Since the keynote paper covers thoroughly the entire range of issues surrounding front end and gateway systems, this paper will concentrate on issues that I see (1) as important to information practitioners and (2) which appear not to be discussed in other papers. Most of the speakers are focussing on the user-computer interaction in the search and retrieval process. But this is only one part of the whole problem, albeit an important and interesting part. In trying to implement this technology in what is generally referred to as the "real world," there are other considerations of equal importance. Some of them are much less interesting problems, actually. But, if the goal is to make systems more usable, by more people, they need to be addressed.

The discussion below is limited to products commercially-available now in the United States for reference/public services applications rather than online catalogs. The topics are viewed from the perspective of both a librarian using these products directly, and a librarian-interpreter/consultant/educator for the so-called end user. Many of the factors are part of the environment of the actual search interface.

2. DIVERSITY & COMPLEXITY

The information retrieval world has grown much more complex since the advent of the various types of automated search intermediaries. In order to utilize this technology in an informed manner, public service librarians and other information specialists must understand a complex universe which consists of:

DATABASES -	2900
VENDORS -	454
GATEWAYS -	35
MICROCOMPUTERS -	200+
MODEMS -	80+
GENERAL COMMUNICATIONS SOFTWARE -	200+
SPECIALIZED COMMUNICATIONS SOFTWARE -	40+

TABLE 1: THE ONLINE UNIVERSE

For most librarian/searchers, microcomputers, modems, and software are more difficult to keep up with than vendors and databases. While one can argue that they should, the reality is that most reference librarian/searchers are not avid readers of InfoWorld who can readily identify com1 and com2, or understand levels of Hayes-compatibility among modems, for example. TABLE 2 lists characteristics of this universe.

MORE DIVERSITY AND COMPLEXITY
INTERFACING COMPATIBILITY PROBLEMS
VERY RAPID CHANGES
HIGH INITIAL COST
LIMITED REVIEW POSSIBILITIES
LIMITED USER SUPPORT
 TRAINING
 HELP LINES
 DOCUMENTATION

TABLE 2: ONLINE UNIVERSE CHARACTERISTICS

What has happened is that systems designed to make searching simpler have at one level--point of choice--made the problem much more difficult. Of particular note is the contrast between the highly-developed support systems of the major online vendors and the difficulties often encountered obtaining support for troubleshooting microcomputer software and hardware problems.*¹ Certainly the excellent vendor support and the widespread availability of training, have been critical factors in the adoption of online systems in libraries.

The online environment has another type of complexity--that of competitive/complementary systems.

FRONT END SOFTWARE
GENERAL COMMUNICATIONS PACKAGES
GATEWAYS
USER-FRIENDLY SYSTEMS
 ONLINE
 CD-ROM

TABLE 3: COMPETITIVE/COMPLEMENTARY SYSTEMS

There is considerable controversy just now as to how CD-ROM will fit in. In theory it should be complementary to online use; in reality, librarians must decide how to allocate limited staff and other resources to support end users using online services,

¹ Two exceptions are Sci-Mate from the Institute for Scientific Information and the software published by Personal Bibliographic Software, Inc. which are well-supported by these institutions.

CD-ROMs, and print products. Compact discs have substantial advantages in some situations involving end users. These are discussed below.

3. TELECOMMUNICATIONS

Research by David Penniman has shown that unsuccessful logons are the most prevalent failure in the use of online search systems. (1) Nearly all microcomputer-based front end packages contain programmed auto-logon sequences, or the capability to allow the user to create such sequences. This is only a partial solution. The problem is that communications are unreliable and fickle. Failure to communicate is not handled well by many software packages and users with no basic understanding of what is happening are easily confused. In a library where online access is provided to end users, much of the librarian's time is taken up by simply helping users logon.

Similarly, if DIALOG or BRS are responding slowly, the end or novice user has no perspective from which to react. This is not a fault of interface design per se, but a problem of at least the same magnitude as the design of the search command interface.

One of the major advantages of CD-ROM is that use of telecommunications and logging on are not necessary. A library can leave the system up and running all the time. The user immediately starts doing something which at the very least provides feedback.

4. ONLINE COSTS

Many of the interfaces labelled user-friendly (often menu-driven) are more expensive to use than command-driven systems because they are slower and therefore take-up expensive connect time. This can be a formidable barrier to use.

Specialized software packages have taken two basic approaches to cost-cutting. There is the "quick and dirty" method exemplified by Search Helper, Wilsearch, Grateful Med and EASYNET whereby a simple strategy is uploaded, searched and a limited number of retrieved references downloaded without intervention by the user. While effective in controlling costs, this approach has the disadvantage of restricting search strategy development and does not take advantage of the interactive nature of online systems.

Facilitating the uploading of pre-created search strategies is another method to control online costs; examples are found in Sci-Mate, Pro-Search, and Dialoglink. For most searches, however, strategy uploading does not result in a large savings.

One might theorize that the easiest way to make an online system user-friendly is to make it free, or fixed fee--or based on some charge mechanism other than connect time. The press of time due to costs, often referred to as the "meter ticking syndrome," has a negative impact on the online user's search behavior. The tremendous success of the first CAS ONLINE academic program, which offered unlimited access in off hours to academic institutions paying a flat monthly fee, is testimony to this theory. CAS ONLINE has a command-driven interface that does not fit the standard definition of user friendly. This is another area in which CD-ROM has an obvious edge in libraries.

5. COST CONTROL

For an academic library offering an end user service, cost control is at least as difficult a problem as training patrons to use a particular search language. The "quick and dirty" systems help with this to some extent by having set rates per search. But, unless the library has a budget to provide free searching, the patron must somehow be "caught" with the documentation of charges just after each search session. Then, the librarian must collect money or initiate a billing cycle with all the problems these processes entail. Password security is always a concern.

Front end and gateway software vendors have been slow to build convenient mechanisms for charging casual users into their systems--the account manager subsystems of DialogLink and Pro-Search not withstanding. These are billing systems to facilitate charge-backs by intermediaries for searches they themselves have performed. Only Prompt Search has a built-in module for billing end users.

6. SET UP

Demonstrations of micro-based front end systems rarely include showing how to set them up. This is sometimes a time-consuming, complicated, frustrating problem. Woe betide the owner of an off-brand modem, even if it is partially Hayes compatible. One has only to attempt to install a few communications programs to appreciate the merits of a true-blue IBM, a Hayes modem and an Epson FX-80 printer. This is a place where a microcomputer-literate end user would have a definite advantage.

7. COMPARISON WITH GENERAL COMMUNICATIONS PACKAGES

It is easier and quicker for many librarians to learn the basics of an additional vendor command language than it is to install and become facile with some front end packages. Many general communications packages on the market today are excellent. They provide for relatively simple set-up of automatic logons,

uploading of strategies or electronic mail, macro set-up for repetitive commands, downloading into an ASCII file and file management. Some of these packages (for example, the Red Ryder shareware program for the Macintosh) will even learn a logon if it is done once and then store it for future use, a capability not found in any of the expensive front end programs I have examined.

For a librarian who routinely uses several online systems, this would mean that a front end program that accessed a limited number of systems or databases would have to offer significant additional advantages to compete and to make the extra cost and learning effort worthwhile.

The ability to read downloaded records into a data management program, manipulate them, and re-format them to a standard bibliographic format is such a significant capability since this opens up the possibility of creating small internal databases relatively easily. SciMate has this feature as do the BiblioLink and Pro-Cite programs from Personal Bibliographic Software, Inc. The accounting modules of DialogLink, Pro-Search or BRS Prompt might be very attractive to librarians in some situations.

On the other hand, end users who have need for only one vendor system, or one database, may well benefit by using a package designed for that system or database. The greater the specialization, the easier it is to provide a simple, effective program.

8. INTELLECTUAL/VOCABULARY ASPECTS OF SEARCHING

Most intermediary information systems have concentrated on the mechanics of search command languages. Only recently have there been programs which provide assistance with what research has shown repeatedly to be the most difficult problem for new searchers--search strategy, particularly vocabulary selection. (2) Some programs such as The Grateful Med and Wilsearch suggest additional search terms from references retrieved. This is an excellent start, but does not go far enough.

9. DISCUSSION

The search interaction itself is often not the weakest link in the online search process chain. The discussion above points out several factors relating to interface design which have nothing to do with the interface itself. But they are part of the present day online environment and bear upon how effective a particular product might become. System designers and researchers will put much effort into the development of search intermediary software in the next few years. For maximum benefit to online users and software producers alike, this total picture should be considered.

REFERENCES

- (1) Penniman, W. David. Modeling and Evaluation of On-line user Behavior. Final Report to the National Library of Medicine. Dublin, OH: OCLC Online Computer Library Center, Inc., June 1982.
- (2) Fenichel, Carol H. "The Process of Searching Online Bibliographic Databases: A Review of Research." Library Research 2:3 (Fall 1980) 107-27.

DIRECTORY OF SOFTWARE PROGRAMS MENTIONED

BIBLIOLINK

PRO-CITE

PRO-SEARCH

Personal Bibliographic
Software, Inc.
P.O. Box 4250
Ann Arbor, MI 48106

SCI-MATE

THE SEARCHER

THE EDITOR

THE MANAGER

Institute for Scientific
Information
3501 Market Street
Philadelphia, PA 19104

DIALOGLINK

DIALOG Information
Services, Inc.

3460 Hillview Avenue
Palo Alto, CA 94304

SEARCH HELPER

Information Access Corporation
404 Sixth Avenue
Menlo Park, CA 94025

EASYNET

Telebase Systems, Inc.
134 North Narberth Avenue
Narberth, PA 19072

WILSEARCH

The H. W. Wilson Company
950 University Avenue
Bronx, NY 10452

GRATEFUL MED

National Library of Medicine
8600 Rockville Pike
Bethesda, MD 20209

PROMPT SEARCH

BRS Saunders COLLEAGUE
1290 Avenue of the Americas
New York, NY 10019

RED RYDER

Free Soft Company
10828 Lacklink
St. Louis, MO 63114

AUTHOR AFFILIATION: Carol Fenichel is Director of the Library at
Hahnemann University in Philadelphia and
Adjunct Professor at the College of
Information Studies, Drexel University.

ADDRESS: Hahnemann University
Broad & Vine
Philadelphia, PA 19102-1192

STRUCTURAL VERSUS APPLICATION KNOWLEDGE FOR IMPROVED DATABASE INTERFACES

Gio Wiederhold

Research within the KBMS project has explored a number of areas where techniques derived from research in Artificial Intelligence can be applied to databases. This research started in 1977 with support from DARPA and has been very productive. The efforts made by the originators and maintainers of databases to provide high-quality databases help these techniques to be effective and reduce the complexity of their application.

Our experiments involve 4 databases:

1. A structured database of ships and ports.
2. A design definition database of the DEC-PDP11 computer.
3. A database of clinical records from an immunology clinic.
4. A bibliographic database of citations and evaluation of the database literature developed in-house, with approximately 5000 citations.
5. The SPIRES library database has been used for one experiment which will be further presented later.

Multiple computer systems are used, and communications are mediated through the ARPANET.

Our research objective has not been to develop a prototype system, but only to develop and demonstrate concepts. We have come to the point now where the techniques available from artificial intelligence no longer suffice. Current research is oriented toward the concurrent interpretation of a knowledge base, directly linked to a database. The knowledge base provides multiple conceptual hierarchies so that the interpreters can be directed to use inference strategies suitable for the task at hand, and even switch strategies when partial goals are satisfied.

Instead of attempting to provide a broad but shallow overview of the research, we have attached the list of 200 papers and reports which have resulted from this research, and a paper which described a library application [Corella 84]. A broad overview of our research is found in [Wiederhold 84a] and a historical overview of structured databases is given in [Wiederhold 84b].

Interfaces, SPIRES, Knowledge-Based Management Systems

A PRECIS OF RESEARCH ON KNOWLEDGE-BASED MANAGEMENT SYSTEMS

June 1986

Investigators:

Professor Gio C. M. Wiederhold
Computer Science Department
Stanford University
Stanford, CA 94305

Professor Jack Milton
Department of Mathematics
University of California at Davis
Davis, CA 95616

Research Participants:

Robert Blum, MD, PhD
Stanford University

David Beech
Hewlett-Packard Research Laboratory
Palo Alto, CA

Prof. Stefano Ceri
Politecnico di Milano
Milano, Italy

Ralph Cherubini
Digital Equipment Corporation
Hudson, MA

Barbara Grosz, PhD
SRI International

Mike Walker (RADIX Project)
Stanford University

Richard Waldinger, PhD
SRI International

Everything Else:

Mary Drake
Computer Science Department
Stanford University

The Knowledge-Based Management Systems Project

* This research has been mainly supported by the Defense Advanced Research Projects Agency under MDA903-77-C-322, N39-80-G-132, N39-82-C-250, and N39-84-C-211. Work on the RX project was funded by the National Center for Health Services Research (HS3650 and HS4389) and continues under sponsorship of the National Library of Medicine (LM4334). Work in natural-language interaction has been partially supported by NSF (IST-8023889). Computer facilities for medically related applications were provided by the Division of Research Resources of NIH. Additional equipment is being provided through a cooperative grant provided by Digital Equipment Corporation, through their Hudson, MA, Artificial Intelligence Research Group. The views and conclusions contained in this document are those of the authors and should not be interpreted as representative of the official policies, either expressed or implied, of any agency of the U.S. Government.

Former Investigators:

Peter Apers, Vrije Universiteit, Amsterdam, The Netherlands

Jerrold Kaplan, PhD, now with Lotus Corp.

Kurt Konolige, PhD, SRI International, Menlo Park

Gordon Novak, PhD, now at Univ. of Texas at Austin

Earl Sacerdoti, PhD, now at Teknowledge, Palo Alto

Daniel Sagalowicz, PhD, now at Teknowledge, Palo Alto

Former Research Participants:

Norm Haas, SRI International

ChoChun Hsu, Beijing University

Prof. Sham Navathe, University of Florida, Gainesville, FL

Prof. Arthur M. Keller, University of Texas at Austin

Ingeborg M. Kuhn, PhD (AAMRS project), now at Veterans Admin., San Francisco

Prof. Robert Paige, Rutgers University, New Brunswick, NJ

ZaiSheng Song, Beijing University

ShiWei Tang, Beijing University

Prof. David Warren, SRI International; now at Manchester Univ., Great Britain

JuanFen Yu, Beijing University

The Knowledge-Based Management Systems Project

Research Assistants:

Most of the credit for the results achieved is due to the students which participated in the project. This list includes research assistants within the KBMS project as well as students who were supported by external funding or by related projects and participated as part of their academic work.

1979-1980	James Allchin
1981	Jim A. Andrews
1981	Mike Anderson
1977-1981	Anne Beetem
1980-1982	Robert L. Blum, MD (RX project)
1981	Stefano Ceri
1984-current	Sang K. Cha
1981	Edward Chan
1985-current	Surajit Chaudhuri
1983-1984	Lei-Hou Chee (CVS)
1981-1982	Francisco Corella
sum. 1980	Lori Craven (Bell OYOC)
1979-1983	James E. Davidson
1982 summer	JinLi Dou
1977-1980	Ramez A. ElMasri
1984-1985	Goran Fagerstrom
1977-1982	Sheldon Finkelstein
1983	Victor Franco (Bell OYOC)
1977-1979	Hector Garcia-Molina
1986-current	Keith Hall
1984-current	Waqar Hasan
1983 summer	Ram Kedlaya
1980-1985	Arthur M. Keller
1979-1982	Jonathan King
1983-current	Linda DeMichiel
1980-1981	Toshimi Minoura (Prof. Susan Owicki)
1983-1984	Robert W. Molter (US Army)
1984	Diep Lan Trinh
1980	Kin-Kee Leung
1985	Kurt Lingel (Amdahl Corporation)
1979-1983	Mohammed Olumi (part. supported by IBM)
1980-current	Edwin Pednault (SRI AI center)
1983-1984	Barbara Pernici
1979-1980	Patricia Pickett (RX project)
1983-current	XiaoLei Qian
1982-current	Peter Rathmann (NSF fellow)
1980-1983	Neil Rowe
1977-1981	Thomas Rogers
1981	Gary Rubin (ARAMIS project)
1984	Luigi Semenzota (H-P)
1979-1980	David E. Shaw
1983-1984	Kitty Shih (RLIN)
1983-current	Mien Shih (part. supported by IBM)
1979-1980	Garrett Short
1984-current	Arun Swami
1980-1983	Kyu-Young Whang
1982-current	Marianne Winslett (Bell fellow)
1984-current	John Woodfill (NSF fellow)
1981	Lena Yesil

The Knowledge-Based Management Systems Project

1. Summary

The Knowledge Based Management Systems (KBMS) Project addresses the problems of intelligent processing in large databases. Such databases are vital to the management functions of modern enterprises. These databases include centralized collections of data as well as data distributed over heterogeneous computer networks. The data is stored on files and maintained by a variety of programs. To make effective use of such data the available *knowledge* about the data has to be formalized in such a way that it can be exploited automatically.

We have investigated a wide variety of types of knowledge about data and databases, from relatively formal semantics based on relational models, to probabilistic and uncertain concepts. The goal of individual investigation is to devise means to make access to data more convenient, more effective, or more efficient. We have also devoted research efforts to the problems arising with the maintenance of databases.

2. Rationale

The utility of computers and computer-based systems in large organizations has been limited more by inadequate techniques for managing large quantities of data than by our technical ability to gather and store those data. Existing systems for collecting, storing, and retrieving data are too inflexible and complex for easy extraction of information. To obtain information, data must be selected and processed so that it can be of value to decision-makers. Traditional output from databases in the form of large reports and messages from the execution of formally stated queries represents primarily selected data rather than information. Typically, several layers of management and technical staff intervene between the data systems and the users, impairing the efficiency of decision-making that depends on information stored within databases.

Managers are aware of the value of control of (and access to) information; they depend on control of the data developed and used by their enterprises. The centralization of this data, which until recently was an essential part of having and sharing large databases, has resulted in a loss of control because of awkward access and dependence on remote personnel. While economic and technological considerations have favored the centralization of data storage and processing during the past, recent directions in hardware development, including VLSI technology, are making available networks of smaller, yet very powerful, special purpose computers that can be used to collect, store, and process information close to its source. Loosely-coupled networks of different machines, personal workstations, etc., are likely to be the dominant type of computing environment within a few years. Consequently, local control of local information will be the rule rather than the exception. Such distributed environments pose new difficulties for the management of data resources. For instance, while control over local data has improved, access to remote data may be more difficult.

Another challenge presented by the move to distributed data distributed systems, is that the knowledge and authority required to manage the diverse sources of information are themselves distributed among multiple sites [Wiederhold 82e]. Familiar methods of data management—in which perfect information about the structure, semantics, and content of

the data is normally assumed—are not adequate to deal with the less structured environment encountered in truly distributed systems. We have found such areas, where knowledge is not perfect, ideal for the application of artificial intelligence techniques.

Developments in artificial intelligence over the last two decades provide tools that can be of value in increasing the responsiveness of database systems to their user's needs. To pursue our work in applying artificial intelligence techniques to the practical domain of database management, we study the problems of data management in centralized and distributed computing environments.

Difficulties occur in every aspect of data management: database design, understanding the information potential of the database, and actual use of the computer. Specific issues include data abstraction, query processing, security, redundancy, optimization, integrity, and decentralized processing.

3. Research Approach

The history of database development and research shows a continuing interaction of industrial efforts. The interplay is often of a type where industrial achievements concentrate on handling larger tasks with adequate performance, and research contributes formalizations and clarifying methodologies [Wiederhold 84b]. We follow that scenario.

Our research probes the quantitative and complexity boundaries of data management. Large quantities of data, complex data relationships, and the limits of storage technology all require that advanced techniques be pursued if realistic problems are to be approached effectively.

In order to understand the problems formally we typically begin by investigating methods and algorithms available in database design and operation [Wiederhold 83b]. We first ascertain the limits of algorithmic approaches and new technology, and subsequently develop heuristic methods to probe beyond those boundaries. In much of our research we subsequently bring to bear the techniques developed in the artificial intelligence (AI) community for dealing with partially understood, arbitrarily structured problem domains [Barr 80], [Wiederhold 83f].

The introduction of heuristics leads to imprecision. We can quote, however, the British mathematician Alan Turing [Hodges 83], who stated in 1946:

In other words then, if a machine is expected to be infallible, it cannot also be intelligent.

The capability of symbols, manipulated through these AI techniques, to represent generalizations and abstractions, provides the basis for intelligent behavior. Recognition of uncertainty, which is an essential aspect of dealing with higher level concepts rather than with specific instances, is hence an essential feature of knowledge in our problem domain.

The specific AI tools are based on such concepts as semantic modeling and build on existing database technology. The databases are effective keepers of information describing the instances that the knowledge is applied to [Wiederhold 86a].

Although the traditional province of database management is in business data processing, we are also considering the data-management of large scale scientific problems [Wiederhold 82g]. The diagram in Section 10 of this report provides a layout of the tasks we have addressed.

To provide an experimental basis for our research we maintain several large databases. One of these includes data on 34 000 merchant ships, ports, and shipping information. A second database contains design specifications for the DEC PDP-11 computer. A third database contains time-oriented data covering many years of clinical observations on arthritic patients [Wiederhold 75]. Finally, we maintain a large bibliographic database of database research references. In April 1984 this file contained citations for 365 books, 1726 refereed papers, 906 proceedings papers, 478 reports and 112 manuals. Of these 1486 include short abstracts.

Most of the early work was carried out on DEC-10 and 20 computers at SRI International and Stanford. For the new tasks, described in Section 5, we have equipment configured to emulate a modern workstation, with 8 megabytes of memory and 600 megabytes of storage, and access to the ethernet. The current computing engine is a VAX 750, with graphic and speech output capabilities. Primary system software is Common Lisp and the RDB relational database under VAX VMS.

A variety of database-management approaches have been used in our research, network [CODASYL 71,74], relational [Codd 70,71], [Blasgen 77], [Held 71], as well as special ones. The medical data are kept on TOD [Weyl 75]. The bibliographic database is available for search in fully inverted form using the news service (NS) system on the Stanford SAIL computer. This bibliography can also be shipped over the ARPA net. A printout can be furnished upon request, but we consider it too large for manual perusal.

We concentrate on promising task areas within our research definition rather than on building complete systems. Since this note is intended principally as a review of our own research, we have not cited all of the many publications of other authors that have provided us with pertinent results, important ideas, and useful guidance in respect to general research directions. General references which contributed significantly to our research are listed in Section 12 of this report, while the papers published under the aegis of the KBMS project are listed in Section 11.

Research Areas and Results

Our research falls into the areas of database design, human interfaces for databases, and performance models of database operation for both centralized and distributed databases. Our work is concentrated at a high conceptual level so that our results may be applied to relational, hierarchical, and network implementations. The application of knowledge about databases is a critical ingredient in all of our research [Wiederhold 84a].

We have categorized our efforts into six areas, although many issues overlap. The six sections (4 through 9) which follow provide short summarizations of our research. Many subsections represent a PhD dissertation and related research. Section 10 presents general conclusions about the results obtained.

- Sec. 4 Design Methodology: The Structural Model; The Data Definition Facility of CRITIAS; Integration of Diverse User Models; Integration of Diverse User Models at Diverse Sites; The Design of Physical Databases from the Integrated Model; Distributed Database Design; Design of Database Management Systems.
- Sec. 5 The Architecture of Knowledge and Databases: Inferencing Over a Database; Formal Semantics of Nulls; Subset Definition; Lexicon Management.
- Sec. 6 Maintenance of Databases: Update Constraints; Updating through Views; Natural-Language Database Updating; Finite Differencing; Maintenance of Distributed Databases; Distributed Resiliency Mechanisms.

Sec. 7 Querying Databases: Graph Model of the Database; Cooperative Interactive Interfaces to Databases; Descriptive Responses to Database Queries; Task Models; Semantic Query Optimization; Common Expression Analysis; Use of a Statistical Abstract of a Large Database.

Sec. 8 Applications: Experiments with Database Technology to Support VLSI Design; Planning; Experiments in Intelligent Data Analysis.

Sec. 9 Tools: File Access System; Intelligent File Systems; Database Machines; Access to Optical Storage.

We will now present these topics in more detail and cite the relevant references. Note that this precis is limited to work actually performed. Work being planned or in progress will be reported in further editions of this precis, as well as in separate publications.

4. Design Methodology

This area concerns research to improve the techniques of database design and modeling. The *structural model*, formally defined within the KBMS project, captures those semantics that are of importance in the design of the database's physical structure [Hendrix 75] [Wiederhold 77,78a,b,82d,83a]. The concepts of the structural model have been expanded and applied in other research aimed at facilitating various stages of the database design process. They provide the basis for the data definition facility of CRITIAS, which implements a schema language based on the structural model [Qian 85b]. They are used in the task of integrating diverse user views, whether in a centralized or distributed setting. And finally, they are used to aid the design of physical databases from the integrated user model.

Further avenues of design oriented investigation have been distributed database design and the design of database management systems.

4.1 The Structural Model

The structural model adds the concept *connections* among relations to the basic concepts of *relations*. Connections model relationships between entities, whereas relations primarily model entities. Connections have formal properties, embodying constraints related to functional and inclusion dependencies. Rules for the maintenance of correctness under update of the database are also given. Three types, *ownership*, *reference*, and *subset* describe distinct constraints imposed by the user on the relationships. The structural model can be considered as having formalized the important relationship aspects of general semantic models. These models (e.g., the entity-relationship models) are used to embody the requirements of the users [Chen 77] [ElMasri 79b] [Shipman 81] [Wiederhold 79a,c,82b,83b].

Our work on database design is based on the use of these structurally relevant semantics, which form the basis for defined connections in the database. Three connection types are adequate to model any structural database constraint other than arbitrary functions: *ownership*, *reference*, and *subset*. When the expected usage loads are mapped to defined connections, the codified knowledge will provide the quantifiable guidance for the physical design. The correctness of binding decisions is necessary to make large databases capable of acceptable performance [Wiederhold 81b] [Ceri 81,85b]. The existence of a model also allows making decisions about reconfiguration of a database as the loads change over time. The problems of asynchronous operation of distributed databases are being modeled using the *identity* connection [[Wiederhold 86e].

It does not matter in what manner the connections are implemented. This permits the logical design to proceed independent of the eventual implementation. This freedom is critical for databases

which are expected to outlive current technologies and for databases which may be implemented differently on multiple sites in distributed system.

The primitives of the structural model are also applicable to databases serving non-traditional functions as engineering information systems for hardware [Wiederhold 82b] and maintenance control systems for software [Olumi 83].

4.2 The Data Definition Facility of CRITIAS

In order to clarify the concepts of schemas independent of their implementation we have developed a free-standing schema definition language CRITIAS. The language provides syntactic embodiment of a semantic data model. The basic modeling concepts of CRITIAS are entity types, attributes, and three types of relationships (subtype, association, and reference), which are intended to model the conceptual objects, their properties, and relationships among them. Language constructs are provided for specifying semantic integrity constraints at all these levels of modeling concepts. We also show that there is a direct mapping from CRITIAS constructs to relational schema hence the implementation of CRITIAS is straightforward [Qian 85b]

4.3 The Integration of User Models into a Comprehensive Database Model

In order to generate a comprehensive database model many data models of individual applications may have to be integrated. The design of a large database must consider many potential users, each with a distinct perception of what the database should look like. The structural model supports the integration of many such views into a database model to support all users [ElMasri 79a,80]. Included in the model is a formalism for the clear expression of structural-integrity constraints in each user's view of the application area and rules for resolving these constraints relative to all users. The integrated database model can remain invariant while performance and operational constraints cause the implemented version of the database to change [Wiederhold 83a,82d]. Each integrated application also obtains access to the data belonging to other applications in a consistent manner. The eventual database submodels for the applications will be richer than their original specification. These revised database submodels can be used to define views and windows for user access to the database.

4.4 Integration of Diverse User Models at Diverse Sites

In a centralized operation a database administrator is responsible for the design of a central database that supports the diverse views of individual users, each of whom can be aware of only a portion of the actual database. In truly distributed situations there may be no single individual with sufficient global knowledge (or authority) to perform this task [Wiederhold 83a]. While no central database will exist, a comprehensive data model is critical for automatic maintenance of overall consistency and for performing distributed processing of queries and updates.

An alternative approach, distributing integrity authority and responsibility to autonomous local sites has been explored in the context of networks of personal computers. The bulk of actual maintenance actions can be delegated to rule-driven agents, which inspect the update messages [Wiederhold 84d].

4.5 The Design of Physical Databases from the Integrated Model

The structural model permits the attachment of projected usage factors from the individual users, and the combined knowledge can be employed to design an effective database implementation. Such a database may be implemented using technology based on relational, hierarchical, or network concepts [Wiederhold 82a].

A straightforward algorithmic optimization approach is not feasible when designing a multi-file database since the combinations of the decision space greatly exceed reasonable time bounds.

Addressing this problem, we have developed a method which will provide implementation guidelines for a relational database design in close to linear time [Whang 81a,b, 83a, 84], [Wiederhold 83a]. The concept developed is termed *separability* of joins between the participating relations.

When the relations of a database are separable, the allocation of clustering and indexing choices for a database implementation can proceed relation by relation. The assumptions required for this approach appear to be fulfilled over a wide range of practical cases. We have also extended this approach to network databases [Whang 82].

A by-product of this research is an improved equation for the estimation of block accesses when retrieving records from files [Whang 85]. This formula has been adapted for the optimization of query processing in relational systems.

4.6 Distributed Database Design

We have also explored the design issues in distributed systems and explored useful algorithms for optimality. In this process we have again found limits of design problem complexity which require the development and evaluation of heuristic approaches. We have developed and published algorithms for the partitioning of databases by selecting fragments defined as tuple sets (horizontal partitioning) or as attribute sets (vertical partitioning) for distribution. In a related project we have developed and evaluated algorithms for distribution within clustered processors.

Our work on database partitioning uses structurally relevant semantics to define connections in the database. These connections and the expected usage loads provide the guidance for the physical binding design decisions [Ceri 81].

This issue maps directly into distributed systems: the cost of connections that go via communication links becomes even more critical. We assume initial user input to select candidate fragments of the database for partitioning and a list of important transactions using those fragments. With each transaction an origin site is associated. A simple model that permits replication but ignores storage and update costs would have that site contain all its needed fragments.

When storage and update costs are considered, the optimal site assignment for the fragments is very difficult. We have been able to formulate and solve an integer programming model for this problem for a modest number of fragments and transactions when there is no replication. We have also analyzed the heuristics required when replication is permitted [Navathe 82], [Ceri 83].

Multiple processors may also be used in clusters, perhaps connected by a high-bandwidth local net, in order to achieve higher aggregate performance than a single large processor can provide. When the processors are clustered, the entry point for a transaction is of little concern. Partitioning is now constrained by limits imposed by processor, input-output, and communication capability [Sacca 83], [Sacca 85].

However, in distributed databases a major technique to achieve acceptable performance is the replication of data [Garcia 78c] [Barbera 82]. We are beginning to model this technique using the concept of an identity connection [Wiederhold 83b]. When replication is used in the centralized environment the binding is implemented quite rigidly (i.e., no access is permitted until updated replicated elements are all set to identical values) [Wiederhold 81b]. We find that the cost of maintaining rigid identity connections is quite high in distributed systems, and are investigating methods that permit a relaxation of those constraints.

In all those approaches to database design, the limits of algorithmic design methods seem to have been reached. The additional complexity that occurs when the fragments can be replicated always requires heuristic approaches [Ceri 81].

When databases are not maintained through a central administrative service, we speak of

a loosely coupled system. These systems will require techniques to deal with the uncertainty of identifying relevant data, locating their instances, and determining their status in terms of currency and quality. We have begun to investigate techniques where expert systems help manage these issues [Apers 84b,c]. While there is an inherent level of uncertainty in the obtained results, it appears that the uncertainty can be quantified and manipulated. Development of adequate techniques in this arena is essential so that the benefits of autonomous systems are not outweighed by the difficulties of sharing data in these systems.

4.7 The Design of Database Management Systems

Understanding the flow of data and the flow of knowledge to control the data has provided the critical underpinnings for the design of a proposed new DBMS. Here we have to consider the information flow in an application-independent manner. We consider all levels of the database system implementation problem starting with a simple operating system interface and going up to the level of knowledge-based user services [Wiederhold 86b].

This research is motivated by a plan to re-engineer a large multi-site control and command system, WWMCCS, using modern software principles. In accordance with Department of Defense (DoD) policies, the software for the WWMCS Information System (WIS) is designed to be implemented in Ada.

The methodology of the specific work is to *start out fresh*. This permits the use of modern software and data engineering approaches to database systems, using the power of Ada in the most appropriate way. Most of us believe strongly that the cost of adapting re-engineered applications to 20-year and 10-year old technology, as exemplified by existing DBMSs, is greater than the cost of rethinking and a new DBMS implementation. If we do not put the principles we have learned in the past twenty years to work now, we will be overtaken by others who do not carry the baggage of the past.

The proposed architecture takes advantage of the structure of Ada. Since Ada supports alternative code bodies with the same declarative specification, it becomes natural to maintain and use alternate modules. However, in order to keep the specification unchanged, the initial design must consider the information requirements by any candidate module. In practice, modules developed in the prototyping stage will not use information needed for an optimizing version of the same module type. Similarly, modules for a workstation database will not use much of the security information available for protection in a shared environment [Wiederhold 83e], [Friedman 86].

The context of this research project reinforced the consideration of security and access controls within the initial design, rather than treating security as an add-on. If a high level of access protection is needed, overall reliability and performance will be improved by incorporating clean and appropriate interfaces in the original system design. Whenever the requirements are less strict, the cost of having null or minimal modules in their place will be small [Spooner 85].

Proposing a fresh start for a major system carries a considerable risk. Doing this in the database area is made more feasible due to the initial orientation of Ada. Since Ada was designed as a language for embedded systems, it did not inherit any prior data-processing notions a problem other general purpose languages such as PL/I have to cope with. Only the most primitive input-output functions are specified in Ada. Hence, we were able to start with a relatively clean slate.

This project presents an example of a data-oriented design methodology. In software projects where we expect that a common data structure will serve many application functions, a data semantics-oriented approach may be a better alternative to a top-down design starting from a procedural objective. In this *data engineering* approach we still proceed top-down, but start from the

information requirements.

Especially when we envisage that modules are replaceable, procedural specifications do not provide a solid base. No procedure can generate information unless the required data and knowledge are made available to the module. When data and knowledge are identified initially, then, in most cases, a competent programmer will have few problems in coding the required transformation procedures and generating a successful program.

On reflection, it appears that the architectural approach proposed, as a conceptualization of architectures found in current DBMS, bears a significant resemblance to the *blackboard concepts* used in some Artificial Intelligence Systems. An early example of a blackboard is found in HEARSAY [Reddy 76] and recent work is BB1 [Hayes 85].

Making the distinction in this design process of:

Knowledge: The controlling and general information, typically complex and relatively small in volume.

Data: The manipulated and factual information, typically regular but voluminous.

helped reveal the design issues and the rules for information management within the design.

5. The Architecture of Knowledge and Databases

As we are becoming increasingly dependent on formalized semantics to handle large databases, we have to be concerned with the architecture of systems that embody both data and knowledge [Malachi 85], [Missikoff 84], [Shuey 86], [Wiederhold 85a,c,86a]. The knowledge bases have to address also issues due to the inherent incompleteness of the recorded facts [Winslett 86], [Blum 82b].

The experience gained in the KBMS research and from other sources is being marshalled for a major subproject, KSYS. The other sources include RX, implementing Knowledge Extraction from Databases [Blum;80,86a,86b]. RX uses frame technology to manage a fairly large database [Wiederhold 81,85a]. Additional experience is derived from INFOBASE, developed at DEC [Cherubini 84] and made available to us.

The KSYS project addresses specifically the need for an effective interface between AI Systems and Database Management Systems [Ceri 86]. The knowledge managed by the AI system represents an enhanced form of the meta-data used to describe databases. The DBMS maintains the facts which represent the continuously changing world and permit new inferences to be drawn by the AI systems linked to the database. We have found, however, that the problems in communicating between AI and database systems are not just in building an interface itself. Each side depends on its own representation structures and inference methods [Wiederhold 86a]. In an integrated system the strengths of each must be exploited. This direction reflects a maturation of the research in this field, where we can go beyond existence proofs and consider quantitative measures and architectural feasibility as well.

The KSYS architecture uses:

- 1) A frame structure for the knowledge base.
- 2) An inheritance scheme which is slot rather than frame based, where each slot is associated with one of an open-ended set of hierarchies.

- 3) A minimal interpreter for knowledge interpretation which can be set to search forwards or backwards.
- 4) An interface from attribute-defining frames to the relational database.
- 5) Persistent storage of all instance information in the database.

A number of research questions are being raised by such an architecture, and will be investigated as resources permit.

5.1 Inferencing Over a Database

When databases are large they can contain implicitly more knowledge about a domain than any single expert. To carry out such inferences, a model of the world represented by the database has to be constructed. Such a model involves abstractions and generalizations, and is hence a knowledge base for that domain [Blum 78,80,82a,c]. We have used a knowledge based system to control statistical procedure which discover and verify new relationships among the abstractions represented in the knowledge base, achieving a form of automated learning [Blum 82a], [Walker 86].

To support the learning process it is necessary to abstract details, stored in the database, to higher-level concepts. By moving to abstractions we can bridge missing data [Blum 81]. If we let the concepts extend over time, we can also start recognizing change, a stronger impetus to action than conventional reports describing a snapshot view of the world [Downs 86].

5.2 Formal Semantics of Nulls

Suppose one wishes to construct, use, and maintain a database of knowledge about the real world, even though the facts about that world are only partially known. In the database domain, this situation arises when database users must coax information into and out of databases in the face of null values and uncertainty. Such incomplete information arises most commonly from missing data in the database, but also through other scenarios such as updates through views. A database containing incomplete information represents a set of alternative worlds, one of which represents the real world, as opposed to the single alternative world modeled by complete information databases.

In the AI domain, the problem of updating a database of incomplete information arises when an agent tries to incorporate new, possibly contradictory information into a database of beliefs reflecting its partial knowledge of the world. In the logic domain, one might choose to represent such a database as a logical theory, and view the models of the theory as possible states of the real world.

How can new information (i.e., updates) be incorporated into the database? For example, given the new information that "b or c is true," how can we get rid of all outdated information about b and c, add the new information, and yet in the process not disturb any other information in the database? The burden may be placed on the user or other omniscient authority to determine exactly which changes in the theory will bring about the desired set of models. But what's really needed is a way to specify an update intensionally, by stating some well-formed formula that the state of the world is now known to satisfy and letting the database management system automatically figure out how to accomplish that update.

Alternatives for semantics of databases with incomplete information and the ramifications of these semantics have been summarized in a paper which won the student award at the IEEE Data Engineering conference in Los Angeles, 1984 [Keller 84a,85b].

We have explored a technique for updating databases containing incomplete information. Our approach embeds the incomplete database and the updates in the language of first-order logic, which we believe has strong advantages over relational tables and traditional data manipulation languages when information is incomplete. We have described semantics and found polynomial-time algorithms

for update operators [Winslett 86,85b]. An implementation of these algorithms has been coded in C. When null values are present in the database or in the updates, incorporating the updates directly in the database may lead to unacceptable expansion in the database size. Using methods reminiscent of those used in concurrency control, we have investigated a lazy evaluation technique which regulates this growth.

5.3 Subset Definition

The subset concept, formalized within the structural model, permits the definition of subrelations which can be used to structurally define classes of entities where data entries would not be appropriate, thus reducing the need for null management. Since use of this concept leads to a large number of conceptual relations, in practice we will map most subsets kept on one processor system into a single file. Control of such mappings is best carried out at a knowledge-based level [Wiederhold 83a].

To provide access to a specific subset, we may define a database view. Use of defined subsets cannot resolve all cases where nulls will occur.

5.4 Lexicon Management

For natural-language systems to provide practical access for database users, they must handle realistic database queries involving terms identifying the stored entities such as names of towns, ships, and people. While formal languages expect the user to ask for, say, "Size of City = 'San Francisco'" this formulation is unnatural. Natural language systems overcome this problem by having all such identifying names within their lexicons, in addition to the common verbs and nouns.

However, databases are often quite large and subject to frequent updates. Both of these characteristics render impractical the encoding and maintenance of a lexicon associated solely with the query language processor, which requires separate maintenance,

We have developed and implemented a technique for reducing the number of lexical ambiguities for unknown terms by deferring lexical decisions as long as possible. The database itself is then accessed to try to resolve the ambiguities and complete the parsing. A simple cost model selects an appropriate method for resolving any remaining ambiguities [Davidson 80].

6. Maintenance of Databases

Before a database can be exploited to yield information, it has to be loaded with data, and these data must be maintained to provide a correct and consistent representation of the real world. The treatment of updates to databases is not yet well treated in theory or practice. Updates to a database may conflict with general integrity constraints, may conflict with previous facts stored in the database, and may involve incomplete knowledge.

Dealing with incompleteness in databases is a major thrust of the KBMS project. It is motivated by the observation that databases, in practice, do not contain data for all data attributes for all stored entities. This is in part an essential result from the concept of sharing data, and cannot be avoided by exhorting users to enter all data. A database pools data from a variety of sources and each source operates under different preconditions and in a different environment. It cannot be expected that a single person or unit will have the global, detailed knowledge necessary to manage the entire database; rather, each specific user of the database will be qualified to manage only a subset of this large collection.

Specific areas of our research have included: view management, natural language or ad-hoc planning updates, subset definition, and as discussed earlier, the effect of null values and other forms of incomplete information.

6.1 Update Constraints

The structural model permits the description of constraints which is implementation independent. Most current systems have only incomplete validation facilities and require that programmers provide procedures which reject inappropriate updates. We have shown that the rules of the structural model provide a basis for efficiently validating updates prior to submitting them to a DBMS [Keller 81a]. If assurance is provided that the rules of the structural semantics have been obeyed at execution time, optimal query paths can then be selected while guaranteeing that the query result will not be affected.

6.2 Updating through Views

The form, content, and availability of a large database are never completely known to any single user. A user is typically restricted to a view or a set of views which define a window onto the database appropriate to the users requirements and authority. This restriction becomes yet stronger when working at remote nodes in a distributed network, since a user at any single site can certainly not be expected to have a global view.

These restrictions lead to a class of problems when the database is to be updated. An update expressed through a view typically has multiple reasonable interpretations in the shared database. For that reason, in most current DBMSs, views can only be used to retrieve information; users operating within a view are not permitted to do updates. This restriction increases the burden on the centralized authority with power to update the database, and introduces errors due to separation of authority over data and authority over database operations.

We have developed a theory which permits enumeration of all possible interpretations of a view update, and a complete collection of algorithms for the implementation of these interpretations. The theory is based on notions of formal correctness of updates [Keller 82,85a,85c].

When the view includes the join columns, view update is generally feasible, especially when the key for the view is entirely contained in one of the underlying relations [Keller 81b]. This work has been extended to include other views formed by standard relational operators (selection, projection). This algorithmic approach is in contrast with the information-theoretic approach based on minimal complements [Bancilhon 81], [Keller 84b].

In addition, we have developed an interactive approach to permit a database administrator to choose the appropriate interpretation for updates at view definition time. Once the interpretation has been fixed, an update expressed against a view will have an unambiguous interpretation in the shared database [Keller 86a].

6.3 Natural-Language Database Updating

Problems conceptually similar to view updates arise in processing natural language queries. The difficulty here is that casual users of a natural-language system do not understand the scope nor the details of the underlying database, and so may make requests that:

- a. are reasonable, given their view of the domain, but are nevertheless not possible in the underlying database;
- b. are ambiguous with respect to the underlying database; or
- c. have unanticipated collateral effects upon the responses to earlier questions or upon alternative views of the database.

The view concept here is a dynamic issue, not subject to predefinition by a database administrator.

Although considerable research has been devoted to the problem of processing queries expressed in natural language, the issues and techniques for performing natural language database

updates have had little attention. The need to perform natural language updates arises when computer databases support ad-hoc planning. When using the database to plan for hypothetical future scenarios, update interpretation should not depend on the insights of a database administrator. For this situation we have developed artificial intelligence techniques to select a plausible update interpretation based on the user's previous interactions with the database, database semantic constraints, and the goal of minimal side effects. If one update interpretation clearly dominates, the user can proceed in the planning scenario without being forced into a tedious disambiguation dialog.

A theory has been developed that makes it possible to identify a particular user's view of the database and to determine the legality, ambiguity, and potential side effects of updates expressed in natural language. A formal analysis of the problems associated with updates expressed on views (database submodels) is central to this work. A system, PIQUE, has been implemented, based on this approach, that processes natural-language updates, explaining problems or options to the user in terms the user can understand and that, moreover, makes changes in the underlying database with a minimal disruption of other views [Kaplan 81a] [Davidson 82,83a,83b].

6.4 Finite Differencing

We have cooperated in the investigation of the transformational technique of finite differencing for the specification, implementation, and application of derived data in the context of a set-theoretic entity-relationship data model. A framework for the automatic maintenance of derived data has been presented. In this framework repeated costly computations are replaced by more efficient incremental counterparts. Applications of this approach are in integrity control and in query and transaction optimization [Paige 82,83a,83b], [Goldberg 83].

6.5 Maintenance of Distributed Databases

The management of redundant, distributed data can seriously affect system performance. We have analyzed and developed algorithms for integrity maintenance [Garcia 77-81]. The use of *hole-lists* to inform nodes of update status of transaction while passing a minimal amount of information has been shown to be effective [Garcia 79e]. The analyses has also shown that it is difficult to improve on the performance of centralized control methods if sufficient backup can be provided in the responsible nodes [Garcia 81].

The separation of read-transactions into three classes, namely those that require no or minimal consistency, those that require auditable consistency, and those that require time-critical consistency, can improve aggregate system performance. The problems arising from serving transactions that support planning functions, which require access to large granules of the database, can be greatly reduced by lowering their consistency demands [Garcia 82].

6.6 Distributed Resiliency Mechanisms

An implied, but rarely realized, benefit of having a distributed database is increased system availability in case of failures. Data availability can be accomplished via replication. Recovery of a node which fails disastrously requires repair and updating of the database at the failed node to achieve consistency.

Such a system is considered resilient with respect to failures, rather than failure-proof. A local node may not be able to provide full local recoverability by itself. This can be due to economic considerations, since the cost of local secure redundant storage can be excessive for small systems, or can be due to operation in a high-risk environment. Techniques to use other nodes within the distributed network to restore failed nodes, whose local storage has been damaged, have been explored and documented [Minoura 78,82,83a,83b]. We are obtaining feedback about their use in financial and manufacturing systems.

Subsequent work in this area deals with a classification of transactions in relation to resiliency. By formally defining a set of transactions which can proceed when the database is partitioned, a set which can be conditionally handled, and a set which cannot proceed, we expect to enhance the operability of distributed databases in conditions of diversity [Apers 84a,d].

7. Querying Databases

To obtain information from a database, it must be queried. High-level interaction with databases by untrained personnel using natural language should be possible. We have applied front-end processors for natural-language queries based on previous work as LADDER and IDA [Sagalowicz 77] and [Hendrix 77].

We have extended the approaches to natural language understanding addressed in earlier work [Gardner 81]. The intent here was to avoid *reinventing the wheel*, so that we can build on research performed earlier and study the many remaining problematical issues in regard to the human interaction with database [Rowe 82c].

When querying issues of incomplete knowledge by the users arise, the structural model can provide guidance in query formulation and in response processing. We also tested the approach to response analysis developed by [Kaplan 79a,b,f,80].

7.1 Using the Graph Model of the Database

The structural and similar semantic models can be exploited to obviate the need for joins for all those connections which are clearly understood. The concepts of ownership, reference, and subset identify ISA hierarchies which can be exploited to locate attributes at higher levels of abstraction. This means that queries giving only attributes, and not relation names, can be understood and processed according to the semantic model, as defined in GORDAS [ElMasri 81]. The computational effort appears to be much less than that required to deduce dynamically those relationships that are based on dependencies in a universal-relation model.

7.2 Cooperative Interactive Interfaces to Databases

In an environment where a user may have incomplete or incorrect knowledge of the structure of available data, queries or updates may be posed that are either unanswerable, impossible to perform, or are otherwise misguided. Our work has shown that such misconceptions can often be detected directly from the phrasing of the user's request and a knowledge of the structure of the database [Kaplan 80]. This information can be used to correct the user's misconceptions and inform the user about the nature of the database and database system. The usefulness of our techniques for cooperative processing of transactions has been demonstrated in a library environment [Corella 84]. These techniques need to be extended to distributed environments, where the knowledge necessary to aid users in use of the database may partially reside at remote sites.

7.3 Descriptive Responses to Database Queries

The typical response to a database query is to list the set of items from the database that satisfy the conditions presented in the query. This list can be excessively long and consequently may be inappropriate for a conversational system. Often, a more appropriate response to such queries is a description of the set, rather than a listing of its elements. For example, the response "All corporate officers" may be more concise and informative in response to "Which employees profit share?" than a list of 1000 names. Practical techniques for producing a significant class of such responses from existing database systems without using a separate world model or knowledge base have been implemented [Kaplan 81b,82a,b].

Information from the structural model can provide useful hints for summarization. Since a referenced entity provides a higher level abstraction of the referencing objects, the typically small set of reference connections can provide the candidate relations for descriptive responses.

7.4 Task Models

Certain entities in a database—be they fields, attributes, relations, or more complicated constructs—have certain probabilities of being required in an interrogation of the database. These probabilities are dependent on the particular task a user may be performing and his or her focus and interests [Grosz 77a,b].

By incorporating a task model into the KBMS, these characteristics can be used in a variety of ways to improve the utility of the system. Four main uses of task models have been explored in a KBMS:

- a. inclusion of relevant information not explicitly requested in the response to a query;
- b. organizing responses so that the more *interesting* items are presented first;
- c. checking for semantic irregularities in the performance of the task; and
- d. prefetching of items and fields that have not yet been requested but are likely to be in the near future [Rowe 82d, 83c] [Davidson 82].

We found that tracking the user's focus dynamically during a single session to be beyond our capabilities.

7.5 Semantic Query Optimization

A request for information can often be formulated in more than one way, depending on knowledge about the subject domain and the ingenuity employed in determining the best access path to the desired information [Martin 77]. A question about all ships currently carrying iron ore, for example, can be answered by looking only at information about bulk ore carriers, assuming that it is known that only bulk ore carriers carry iron ore.

Semantic query optimization is an approach to improve the performance of information requests that uses such domain knowledge automatically. The objective is to transform a query into a semantically equivalent one (i.e., one that produces the same answer but can be processed more efficiently).

This work demonstrates that semantic knowledge about the database can be used in a new way to achieve efficient data retrieval. The method supplements conventional query optimization methods. Drawing on semantic, structural, and processing knowledge, it makes use of the heuristics of query processing developed in prior research. A system, QUIST, has been implemented to explore this approach. Improvements have been demonstrated in a range of query types by means of transformations that add or delete constraints on attributes or even entire files, as warranted by the database and query structure. In other cases, the system detects unsatisfiable query conditions without inspecting the database, or aborts inference when it determines that no improvement can be expected. Analysis overhead is low in all cases [King 79,80a,b,81a,b].

7.6 Common Expression Analysis

Another optimization task depends on expressions and their results from prior queries. We have shown that common expression analysis can be applied in three different settings for optimization of database request. The query graph representation of queries was originated for this work. It represents the structure of requests in an intuitively clear manner. Also the decomposition into nodes (relation occurrences in queries labelled with internal selection predicates and projected at-

tributes) and edges (join predicates) serves as a convenient structure for automatic analysis in query optimization.

We have written a program that detects whether or not the result of one query can be used to support answering another query. Finkelstein describes both the query graph and the first two settings for optimization described below [Finkelstein 82].

1. In an interpretive system, answers to previous queries, or temporaries formed in the process of obtaining them, may significantly reduce processing time for subsequent queries. We have defined a methodology, based on the query graph representation, for deciding cheaply whether such temporaries are helpful.
2. A transaction which contains multiple queries may perform redundant database operations. We have described a procedure for optimization of a collection of queries, which involves computing least upper bounds on query graph nodes (which correspond to scans through relations which might profitably be combined). These are extended to maximal subexpressions between queries, and a heuristic procedure examines possible query revisions to find the best execution procedure.
3. We have extended the concept of optimization of generated code to languages with embedded queries (such as SQL in PL/I). We have analyzed how queries in loops can be optimized differentially. Also, we have studied how global flow analysis can be used to determine available database expressions at places in the program.

This work is now being applied within the IBM San Jose Research Laboratory to prototype relational database systems.

7.7 Use of a Statistical Abstract of a Large Database

The size of data sets subjected to statistical analysis is increasing as computer technology becomes more sophisticated [Wiederhold 84c]. It is an attractive alternative for analysis to have quick estimates of descriptive statistics rather than exact values obtained with considerable delay. We have demonstrated a new technique for estimating database statistics that is a top-down alternative to the bottom-up method of sampling. This approach precomputes a set of general-purpose database statistics (a database abstract) and then uses a set of approximately 400 inference rules to make bounded estimates of other, arbitrary statistics requested by users.

The inference rules comprise a new example of an artificial-intelligence expert system. There are several important advantages of this approach over sampling methods, as has been demonstrated experimentally in estimating a variety of statistics on two different databases [Rowe 81b,82b,d,e,f,83a,b,d].

8. Applications

In order to gain experience with our concepts and validate their applicability, we also carry out research in application areas. We have obtained databases of merchant ships, of computer circuits, and of chronic disease patients to support this work.

8.1 Experiments with Database Technology to Support VLSI Design

The implementation of a large and complex device requires a massive design effort. The length of the design cycle of VLSI non-regular devices is often critical. For a complex device, approaches involving only a single designer are either too slow or make so many compromises that they utilize chip area very poorly. If instead, a team of designers work together on a project, they will require much support for communication and task scheduling. We have applied our techniques to manage

databases to aid the VLSI design process. The database is used here as a communication medium among people working at different levels on the same object, namely the same device or system. This use of a database is in contrast to current design automation aids which use task specific files without automatic feedback of analysis results or decisions made by the designers.

Logic and circuit design information of two devices, an ALU and a PDP-11 CPU, and the component library required to build them, have been placed into databases. The lower level data elements can be automatically generated using an existing design automation program. Performance measurements indicated only a performance degradation by a factor of about 2 versus use of specialized design files [Wiederhold 80b]. Modification of lower level elements during the design process is signalled automatically, using a height-first algorithm, to the related parent levels, so that this detailed knowledge can be incorporated in the higher level abstraction when these are accessed during successive design iterations [Beetem 82], [Wiederhold 82b].

We have developed some access techniques for the design data which permit retrieval of data that is partially stored. If required elements are not stored, they are generated from higher level circuit or logic specifications. Partial storage is important. The volume of lower level elements in a VLSI design can be greater than is manageable by the largest storage devices now available, so that automatic methods for VLSI design automation will need access to a mix of generatable, regular elements and instantiated, irregular elements if they are to handle large circuits that are not totally regular [Wiederhold 82b].

Our initial experiments utilized a CODASYL database system, DBMS-20. We are in the process of evaluating our approaches for other database environments to make our results more accessible to practitioners. We have ported the database to a relational and object oriented system at XEROX Palo Alto Research Center and evaluated the performance of the VLSI-design programs in both a purely relational and an entity-relationship model implementation [Winslett 84,85a]. The entity-relationship model improved performance, but the overriding factor in both versions was the cost of writing the results to disk.

An important byproduct of this work is an assessment of the tradeoffs among database implementation models. Much argument has been created in the arena of relational versus network implementations. We believe that some solid experiments in this one critical area can help in quantizing the issue in an objective fashion [Milton 83].

We have begun to consider the issue of the automatic management of design versions. This task is performed now by databases that support engineering change control [Olumi 83]. The same problem is being addressed in conventional databases that are concerned with time or planning updates, but there are yet few results to draw upon, so that in fact the design area may be able to lead in this research field [Belady 77] [Bennett 82] [Parnas 76] [Rochkind 75]. The high capacity permanent storage provided by optical disks permits a greater retention of past versions [Rathmann 84].

8.2 Planning

Databases are important resources for planning. Much planning is based on extrapolation from past events and includes hypothetical data defining alternate future scenarios. The intrinsic complexity of planning achievement of a goal from a current state is high.

In research performed at the SRI International AI Center theoretical issues in planning are being investigated. A method which permits plans to be created, although not necessarily executed, efficiently is the focus of current research [Pednault 84]. The method can be related to many techniques which had a heuristic base.

8.3 Experiments in Intelligent Data Analysis

If database analysis can deal effectively with both past facts and multiple future scenarios, its use as a planning tool will be greatly enhanced. There remains a need for improving the corresponding analytical support capability of planning systems.

Many of the critical data which are found in large systems appear in a time sequence. Our existing systems have not dealt well with the special demands stemming from the temporal relationships among the data. In related projects (ACME, AAMRS, ARAMIS, RX) we have had the opportunity to structure, collect, and analyze large medical databases. In such databases, for instance TOD, time is a key attribute [Weyl 75]. Some of this work was performed at a system level [Wiederhold 80a,c, 81a,c, 82c,f] [Kuhn 82,84], while other work analyzes the content statistically [Blum 81].

Content analysis requires large collections of well maintained data. Many years of operational use were required to get to this stage. In this research we have developed tools that use artificial intelligence techniques to assist in the processing of time-oriented data. A frame-based model is used to capture domain knowledge. This model can assist in combining the database for unexpected time-lagged correlations and report such events as prospective hypotheses. If analysis of the finding appears warranted, a statistical model is built that includes all possible covariates given in the knowledge base.

The appropriate statistical tests are selected by applying a collection of rules that use the descriptions of the data from the database schema and the conditions that are satisfied by the statistical procedures. If the findings turn out to be significant, they are added to the knowledge base for use in another iteration of hypotheses generation and testing. This approach to knowledge generation complements the familiar, but less formal approach taken by expert systems [Shortliffe 79,83???].

This approach has been demonstrated by using a small subset of a time-oriented rheumatology database (ARAMIS) [Fries 79]. The statistical methods now available are limited to multi-variate regression. Tentative hypotheses were generated and automatically tested for statistical significance. Several verified hypotheses could be added to the original loaded knowledge base [Blum 82a,b,c].

Work is continuing in the medical arena, but its applications to other databases (e.g., military manpower management) are also under way. Basic research to deal with reduction of the inherent complexity of hypotheses generation is continuing.

9. Tools

As part of the research we have developed software tools. Some of these have had broader applicability and have been used as prototypes for commercial and industrial database modules. We are further investing in this area again, partially so that we may rapidly build demonstration systems, but also to demonstrate that future database and knowledge base systems can be effectively assembled from standard modules.

Modules developed and under consideration include a symmetric file access system, a database definition language processor, a relational algebra processor, and recovery support. We hope to integrate these on the NAXOS system, a VAX 750 dedicated to experimentation with advanced databases.

9.1 File Access System

A file access system that uses symbolic keys to access variable length records based in PASCAL and supporting several host languages, including PASCAL, INTERLISP, and FORTRAN has been

developed and tested. The services that this system, named FLASH, expects from the underlying operating system are limited to:

1. directory management for named segments of secondary storage, and
2. access to fixed size blocks or pages of these segments.

In a multi-user environment some locking facilities are also needed. Since this subproject is the basis for long-range database system development, reliability and efficiency have been major design and implementation objectives. It is specifically designed to provide strong and symmetric support facilities for databases, so that powerful database systems can become easier to implement than they are using conventional files designed with only programmer's needs in mind. The underlying structure uses B+ trees for storage of both primary and secondary keys [Allchin 80]. This system will be used to study various dynamic storage and retrieval strategies. The experience of implementing FLASH has been used to define an Input-Output package for the Ada language [Keller 86c].

While traditionally the notions of primary key have implied both a high degree of locality as well as uniqueness of key, it is desirable for a database-driven system not to make that distinction. Included in that development is a port to DEC VAX equipment. The available UNIX software is particularly deficient in the area of file management.

9.2 Intelligent File Systems

The understanding we have developed in the file system area leads us to consider hardware implementations for these facilities. The processing capability of modern disk controllers is such that a substantial fraction of a file system could be provided outboard of the main processors.

Such an intelligent file server would respond to high level queries. Whereas conventional equipment requires the processor to specify a device and a storage address for retrieval, we foresee having to submit only a symbolic name and a key. The objects to be retrieved will no longer be fixed-length blocks, but sets of variable length records.

The difficulties of moving advanced file functions out of the processors realm are in the re-specification of the interface. Linkages for serial access within the retrieved sets, algorithms for memory allocation and for setting and managing locks for concurrent access all need analysis and formalization.

9.3 Database Machines

One focus of research activity within the KBMS project has involved the design and analysis of alternative hardware machine architectures oriented toward the very rapid execution of the relational operations which arise frequently in large-scale database management applications. This research has yielded certain surprising and potentially important results which may ultimately prove useful in designing cost-effective, extremely high-performance database machines [Shaw 79].

These results have led to a "high-level" design for a specialized non-von Neumann machine, portions of which would be implemented in VLSI, which would support the highly efficient evaluation of the most computationally complex operators of a relational algebra, specifically projection with elimination of duplicate tuples and the equi-join. Based on a hierarchy of associative storage devices, this architecture permits an $O(\log n)$ improvement in time complexity over the best known evaluation methods for these operators on a conventional computer system, without the use of redundant storage, and using currently available and potentially competitive technology. In many cases of practical import, the proposed architecture should also permit a very significant improvement (by a factor roughly proportional to the size of the primary associative storage device) over the performance of previously implemented or proposed database machine architectures based on associative

secondary storage devices [Shaw 80a,b,81]. This work is the basis for further VLSI data-oriented architectural research in progress at Columbia University.

9.4 Access to Optical Storage

Optical storage technology promises to greatly increase the capacity and reduce the cost of data systems. Optical storage comes with media which are either:

1. created from master disks and can only be read subsequently
2. written once by appending blocks of data, but never erased
3. erased and rewritten, similar to magnetic storage .

The type of optical storage which we consider in particular uses optical disks which are writable once, but can be read often. It is these devices that can serve data-processing needs because of their incremental writability and their high storage density.

While research on rewritable optical media is progressing, it seems that the linear density will be reduced by at least a factor of 2, leading to reduction in storage capacity by a factor of 4.

With this development, new problems have to be addressed. At this time we recognize that:

1. more data will be replicated because storage costs are dropping more rapidly than communication costs
2. more versions of past data will be kept
3. the index mechanisms used for smaller, rewritable devices are not directly applicable to this medium.

While we are addressing the problem of replication in general, the problems of indexing and access to versions has our particular attention. In our databases we find that the space occupied by indexes can exceed the storage space used for the data. Traditional indexing techniques depend on wholesale or incremental update of the index trees. This means that escaping to magnetic storage for index maintenance will create other restrictions. Either past versions will not be indexed, so that they are lost for practical purposes, since scanning of these high volume storage devices will take many hours, if not days; or the granularity of indexing will be high, so that one retrieval must fetch millions of characters, which must then be post-processed.

We have developed a technique where indexing structures, in particular tries and various kinds of trees [Knuth 73], are stored incrementally within the data blocks as they are rewritten [Rathmann 84]. This appears to solve the basic problem, although further investigations are foreseen. The trie approach has been implemented [SYTH 85], in an optical version of FLASH, a language independent portable file access system [Allchin 80a].

10. Conclusion

Our work on Knowledge-Based Management Systems, the Management of Distributed Knowledge, and Management of Knowledge and Databases at Stanford and SRI International has demonstrated that substantial improvements in the utility of database systems result from the application of techniques borrowed from Artificial Intelligence. This work also incorporated considerations of distributed semantics. The incorporation of higher-level semantic knowledge into database systems has been shown to help with the design, improve the manageability, and increase the utility of data in a variety of ways.

Upgrading databases from passive systems which mainly store information to active systems which understand information, by incorporating knowledge of the data and the

domain into the system itself, is a natural and necessary precondition for growth in such systems.

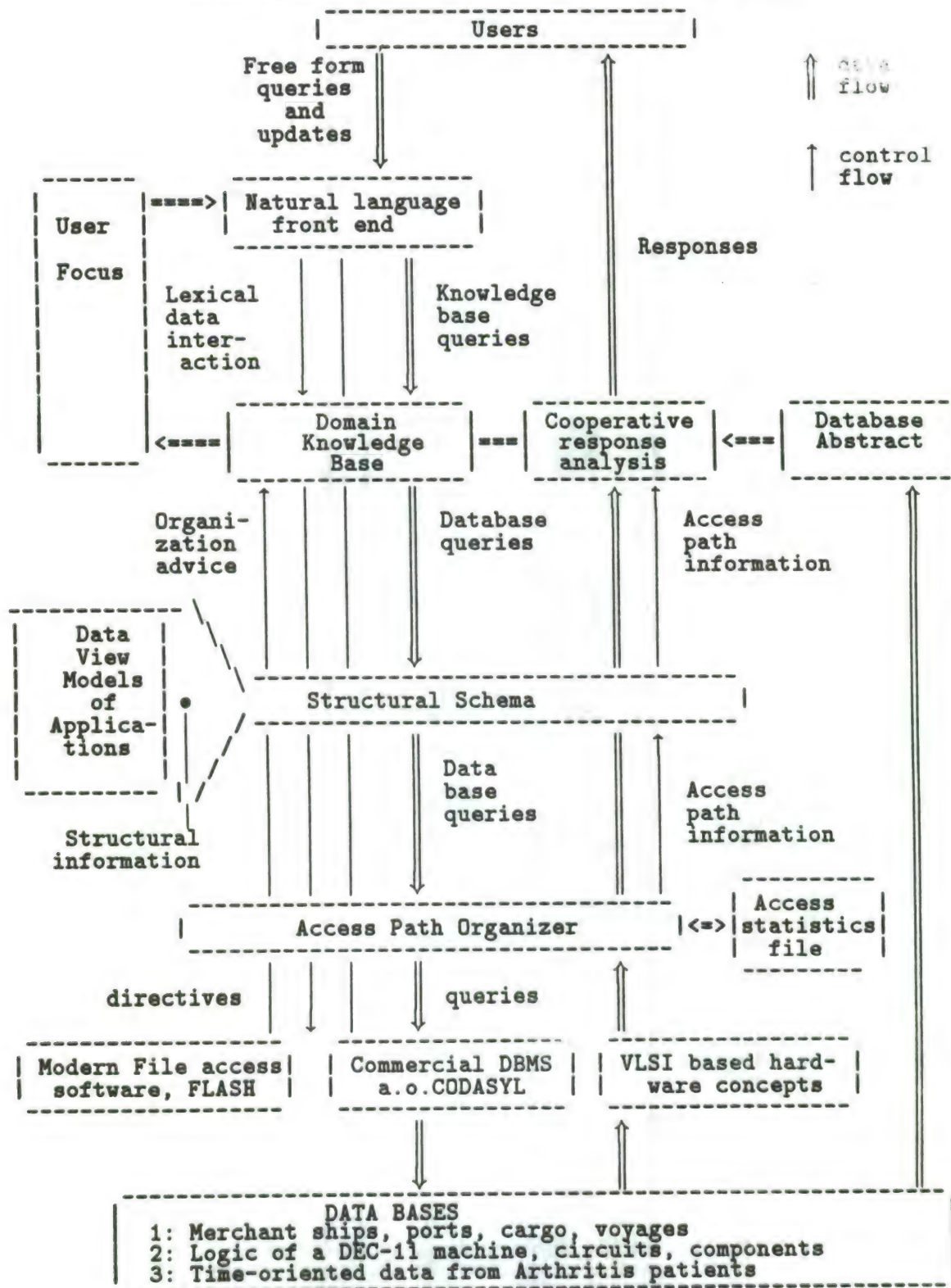
Our research conclusions can be summarized as follows:

- Traditional, algorithmic techniques provide a solid foundation for work in databases, but the models used cannot cope adequately with the complexities and the scale of systems of realistic proportions.
- The techniques developed in artificial intelligence research apply well to the database world because of the intrinsic formal abstraction provided by databases in their modeling of the real world. Artificial intelligence research has been most effective where it has focused on highly structured domains that are subject to simplified abstract descriptions. Databases provide a realistic example of such a domain.
- Within the KBMS framework we have been able to develop innovative approaches to modeling the database needs of diverse user groups and integrating various user database models; improving query processing through the use of task models, semantic query optimization, common subexpression analysis, and giving descriptive responses; as well as dealing with issues of updating large databases, including problems arising due to incomplete data, incomplete knowledge in the users view, and natural-language database updates.
- To support this research we have developed tools, such as portable file-access systems, database technology to support VLSI design, designs for database machines, and methods for optimal design of centralized and distributed physical databases.
- Our work is finding applications in the design and operation of information systems in the world outside the laboratory. Moreover, it is beginning to contribute to tasks being undertaken by NASA, organizations involved in military manpower planning, and other institutions, including applications for manufacturing, medical, economic planning, and financial systems [Ghosh 79], [Wiederhold 79d].

Our work has been characterized throughout by a conviction that future systems will have to deal with large data and knowledge bases. In such environments a user has only a limited view from which to manage the database and infer new information for decision-making. Views may be physically related to distribution of storage.

We are now synthesizing and formalizing the results from our studies. To integrate the concepts developed we have chosen an approach which permits distinct maintenance of data and knowledge, while at the same time integrating the inference processes. To store the knowledge we are employing frames, while the databases uses a relational DBMS. Efficiency issues are being addressed through dynamic binding of information as it is being brought into memory.

Conceptual Interactions of KBMS Components



11. Stanford Reports and Papers

These books, papers and reports document research performed fully or partially under current and preceding ARPA contracts to the investigators collaborating in the KBMS project.

[Allchin 80a]

J. Allchin, A. Keller, and G. Wiederhold: "FLASH: A Language-Independent Portable File Access System"; *Proceedings of the ACM-SIGMOD Conference*, May 1980, pp.151-156.

[Allchin 80b]

J. Allchin: "Modula and a Question of Time"; *IEEE Transactions on Software Engineering*, vol.SE-6 no.4, July 1980, pp.390-391.

[Apers 84a]

P.M.G. Apers and G. Wiederhold: "How to Survive a Network Partition"; submitted for publication, April 1984.

[Apers 84b]

Peter M.G. Apers, and Gio Wiederhold: "Expert Database System in a Loosely Coupled Environment"; *Proc. First Workshop on Expert Database Systems*, Kiawah Island, South Carolina, Oct.1984, Institute of Information Management, Technology and Policy, Univ. of South Carolina, vol.2, pp.611-617.

[Apers 84c]

Peter M.G. Apers, and Gio Wiederhold: "Transaction Handling in a Loosely Coupled Environment"; *Proc. ACM Int. Conf.*, Florence Italy, 1985, North-Holland Pub.

[Apers 84d]

Peter M.G. Apers, and Gio Wiederhold: "Transaction Classification to Survive a Network Partition"; Stanford CS report STAN-CS-85-1053, August 1984; submitted to ACM TODS, August 1984.

[Arvidson 85]

Raymond Arvidson, Gio Wiederhold, et al: *Issues and Recommendations Associated with Distributed Computation and Data Management Systems for the Space Sciences*, volume 2; Committee on Data Management and Computation, Space Sciences Board, National Academy of Sciences, January 14, 1985.

[Banks 85]

Peter M. Banks (chairman SESAC Task Force on Scientific Uses of the Space Station): *Space Station Summer Study Report*; Michael Wiskerchen and Gio Wiederhold: "Section 52: IOC Facility and Operations Requirements: Communications and Information Systems"; NASA, Washington DC, March 21, 1985.

[Barr 80]

A. Barr and J. Davidson: *Representation of Knowledge*; Stanford University CS Report CS-80-793, March 1980; also in *The Handbook of Artificial Intelligence*, vol I, A.Barr and E.Feigenbaum (eds), 1981.

[Beetem 82]

Anne Beetem, Jack Milton, and Gio Wiederhold: "Performance of Database Management Systems in VLSI Design"; *IEEE Database Engineering Bulletin*, vol.5 no.2, June 1982, pp.15-20.

- [Blum 78]
R.L. Blum and G. Wiederhold: "Inferring Knowledge from Clinical Data Banks Utilizing Techniques from Artificial Intelligence"; *Proc. 2nd Symp. on Computer Applications in Medical Care*, Nov.1978, Washington DC, IEEE, pp.303-307.
- [Blum 80]
R.L. Blum: "Automating The Study of Clinical Hypotheses on a Time-Oriented Database: The RX Project"; *MEDINFO 80*, Lindberg and Kaihara(eds.), IFIP, North-Holland, 1980, pp.456-460.
- [Blum 81]
R.L. Blum: "Displaying Clinical Data from a Time-Oriented Database"; *Computers in Biology and Medicine*, vol.11 no.4, 1981.
- [Blum 82a]
R.L. Blum; "Discovery, Confirmation, and Incorporation of Causal Relationships from a Large Time-Oriented Clinical Database: The RX Project"; *Computers and Biomedical Research*, vol.12 no.2, pp.164-187, 1982.
- [Blum 82b]
Robert L. Blum: *Discovery and Representation of Causal Relationships from a Large Time-Oriented Clinical Database: The RX Project*; Lecture Notes in Medical Informatics, no.19, Lindberg and Reichertz, (eds.), Springer-Verlag, New York, 1982, 242 pp.
- [Blum 82c]
Robert L. Blum and Gio C.M. Wiederhold: "Studying Hypotheses on a Time-Oriented Clinical Database: An Overview of the RX Project"; *Proc.of the 6th Symposium on Computer Applications in Medical Care*, Oct.30-Nov.2 1982, Washington, DC, IEEE 82 CH1805-1, pp.725-735.
- [Ceri 81]
Stefano Ceri, Shamkant Navathe, and Gio Wiederhold: *Optimal Design of Distributed Databases*; Stanford CS Report STAN-CS-81-884, Dec.1981.
- [Ceri 83]
Stefano Ceri, Shamkant B. Navathe, and Gio Wiederhold: "Distribution Design of Logical Database Schemas"; *IEEE Transactions on Software Engineering*, vol. SE-9 no.4, July 1983, pp.487-563.
- [Ceri 84a]
Stefano Ceri and Guiseppe Pelagatti: *Distributed Database Design*; McGraw-Hill, April 1984.
- [Ceri 84b]
Stefano Ceri, Barbara Pernici, and Gio Wiederhold: "An Overview of Research in the Design of Distributed Databases"; in *IEEE Database Engineering Bulletin*, vol.7, Dec.1984.
- [Ceri 85]
Stefano Ceri, Barbara Pernici, and Gio Wiederhold: "Design Support Environment for Distributed Databases"; Submitted for consideration, *IEEE TSE*, (W.Chu) March 1985.
- [Ceri 86]
Stefano Ceri, Georg Gottlob, and Gio Wiederhold: "Interfacing Relational Databases and PROLOG Efficiently"; Proceedings of the First International Conference on Expert Database Systems, April 1986; to be republished by Benjamin Cummins.

- [Corella 84]
 Francisco Corella, S.J. Kaplan, G. Wiederhold, and L. Yesil: "Cooperative Responses to Boolean Queries"; *Proc. IEEE Data Engineering Conference*; April 1984, Los Angeles CA, pp.77-93.
- [Davidson 80]
 J. Davidson and S.J. Kaplan: "Parsing in the Absence of a Complete Lexicon"; *Proceedings of the 18th Annual Meeting of the Association for Computational Linguistics*, Philadelphia PA, June 19-22 1980, pp.105-106.
- [Davidson 82]
 Jim Davidson: "Natural Language Access to Databases: User Modeling and Focus"; *Proceedings of the CSCSI/SCEIO Conference*; 1982, Saskatoon, Saskatchewan, Canada, May 17-19, 1982, pp.204-211.
- [Davidson 83]
 Jim Davidson and S.J. Kaplan: "Natural Language Access to Databases: Interpreting Update Requests"; in *American Journal of Computational Linguistics*, April-June 1983, pp.57-68.
- [Davidson 84a]
 Jim Davidson: "A Natural Language Interface for Performing Database Updates"; *Proc. IEEE Data Engineering Conference*; April 1984, Los Angeles CA.
- [Davidson 84b]
 Jim Davidson: *Interpreting Natural Language Database Updates*; Ph.D. dissertation, Stanford CS Report to appear, 1984.
- [Downs 86]
 S.M. Downs, M.G. Walker, and R.L. Blum: *Automated summarization of on-line medical records*; Stanford Memo KSL-86-6, January 1986; to appear in *Medinfo'86*.
- [ElMasri 79a]
 R. ElMasri and G. Wiederhold: "Database Model Integration Using the Structural Model"; *Proceedings of the the ACM-SIGMOD Conference*, Bernstein(ed.), Boston, MA, June 1979, pp.191-198.
- [ElMasri 79b]
 R. ElMasri and G. Wiederhold: "Properties of Relationships and Their Representation"; *Proceedings of the 1979 NCC*, AFIPS vol.49, Aug.1979, pp.319-326.
- [ElMasri 80]
 Ramez A. ElMasri: *On the Design, Use, and Integration of Data Models*; Ph.D. dissertation, Stanford CS Report CS-80-801, June 1980.
- [ElMasri 81]
 Ramez ElMasri and Gio Wiederhold: "Formal Specifications of GORDAS: A High-Level Query Language for Graph Databases"; *Proceedings of the Second International Conference on the Entity-Relationship Approach to System Analysis and Design*, Washington DC, Oct.12-14, 1981.
- [Finkelstein 82]
 Sheldon Finkelstein: "Common Expression Analysis in Database Applications"; *ACM-SIGMOD International Conference on Management of Data*, Orlando FL, June 2-4 1982, pp.235-245.

[Friedman 85]

F. Friedman, A. Keller, J. Salasin, D.L. Spooner, and Gio Wiederhold: "Reference Model for ADA Interfaces to Database Management Systems"; Second IEEE Computer Society Data Engineering Conference, Los Angeles CA, February 1986.

[Garcia 77]

Hector Garcia-Molina and Gio Wiederhold: "Application of the Contract Net Protocol to Distributed Data Bases"; Stanford University Heuristic Programming Project paper HPP-77-21, April 1977.

[Garcia 78a]

Hector Garcia-Molina: "Distributed Database Coupling"; Stanford University Heuristic Programming Project paper HPP-78-4, March 1978.

[Garcia 78b]

Hector Garcia-Molina: "Performance Comparison of Update Algorithms for Distributed Databases, Part I"; Technical Note 143, Stanford University, Computer Systems Laboratory, Departments of Electrical Engineering and Computer Science, June 1978.

[Garcia 78c]

Hector Garcia-Molina: "Distributed Database Couplings"; *Third USA-Japan Conference Proceedings*, AFIPS, San Francisco CA, Oct.1978, pp.75-79.

[Garcia 78d]

Hector Garcia-Molina: "Crash Recovery in the Centralized Locking Algorithm"; Technical Note 153, Stanford University, Digital Systems Laboratory, Departments of Electrical Engineering and Computer Science, November 1978.

[Garcia 78e]

Hector Garcia-Molina: "Performance Comparison of Two Update Algorithms for Distributed Databases"; *Proceedings of the 3rd Berkeley Conference on Distributed Data Management and Computer Networks*, August 1978, pp.108-119.

[Garcia 78f]

Hector Garcia-Molina: "Performance Comparison of Update Algorithms for Distributed Databases, Part II"; Technical Note 146, Stanford University, Computer Systems Laboratory, December 1978.

[Garcia 79a]

Hector Garcia-Molina: "Restricted Update Transactions and Read-Only Transactions"; Technical Note 154, Stanford University, Computer Systems Laboratory, January 1979.

[Garcia 79b]

Hector Garcia-Molina: "Partitioned Data, Multiple Controllers, and Transactions with an Initially Unspecified Base Set"; Technical Note 155, Stanford University, Computer Systems Laboratory, February 1979.

[Garcia 79c]

Hector Garcia-Molina: "A Concurrency Control Mechanism for Distributed Databases Which Uses Centralized Locking Controllers"; *Proceedings of the 4th Berkeley Conference on Distributed Data Management and Computer Networks*, August 1979, pp.113-124.

[Garcia 79d]

Hector Garcia-Molina: "Centralized Control Update Algorithms for Distributed Databases"; *Proceedings of the 1st International Conference on Distributed Processing Systems*, October 1979.

- [Garcia 79e]
Hector Garcia-Molina: *Performance of Update Algorithms for Replicated Data in a Distributed Database*; Ph.D. dissertation, Stanford University, Computer Science Department report CS-79-744, 1979.
- [Garcia 80]
Hector Garcia-Molina and Gio Wiederhold: "Read Only Transactions"; Stanford University Computer Science Department report CS-80-797, April 1980.
- [Garcia 81]
Hector Garcia-Molina: *Performance of Update Algorithms for Replicated Data in a Distributed Database*; (revised) UMI Research Press, Ann Arbor MI, ISBN 0-8357-1219-2, Aug. 1981, 320pp.
- [Garcia 82]
Hector Garcia-Molina and Gio Wiederhold: "Read-Only Transactions in a Distributed Database"; *ACM Transactions on Database Systems*, vol.7 no.2, June 1982, pp.209-234.
- [Gardner '81]
Anne Gardner, Terry Winograd, and Jim Davidson: "Natural Language Understanding"; in *The Handbook of Artificial Intelligence*, vol. I, A. Barr and E. Feigenbaum (eds), 1981; also Report No. STAN-CS-79-754, Computer Science Department, Stanford University, 1979.
- [Gottlob 86]
George Gottlob: "Subsumption and Implication," to appear in *Information Processing Letters*, 1986.
- [Ghosh 79]
Sakti Ghosh, A.F. Cardenas, I. Mijares, and G. Wiederhold: "Some Very Large Data Bases in Developing Countries"; *5th International Conference on Very Large Databases*, Rio de Janeiro, Brazil, Oct. 1979, pp.173-182.
- [Goldberg 83]
A. Goldberg and R. Paige: *Stream Processing*; Rutgers University, Technical Report LCSR-TR-46, Aug. 1983.
- [Kaplan 79a]
S.J. Kaplan: "Cooperative Responses from A Portable Natural Language Data Base Query System"; Ph.D. dissertation, University of Pennsylvania, July 1979; also Stanford University Heuristic Programming Project paper HPP-79-19.
- [Kaplan 79b]
S.J. Kaplan, E. Mays, and A.K. Joshi: "A Technique for Managing the Lexicon in a Natural Language Interface to a Changing Data Base"; *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, Tokyo, Japan, August 1979.
- [Kaplan 80]
S.J. Kaplan: "Appropriate Responses to Inappropriate Questions"; to appear in *Formal Aspects of Language and Discourse*, Joshi, Sag and Webber(eds.), Cambridge University Press, 1980.
- [Kaplan 81a]
S.J. Kaplan and Jim Davidson: "Interpreting Natural Language Updates"; *Proc. 19th Annual meeting*, Association for Computational Linguistics, June 1981.

[Kaplan 81b]

S. Kaplan and D. Ferris: "Hello, This is Your System Speaking"; *Computing*, vol.9 no.43, Oct.22, 1981.

[Kaplan 82a]

S. Jerrold Kaplan: "Cooperative Responses from a Portable Natural Language Database Query System"; in *Computational Models of Discourse*, Brady(ed.), MIT Press, 1982, pp.167-208.

[Kaplan 82b]

S. Jerrold Kaplan: "Cooperative Responses from a Portable Natural Language Query System"; *Artificial Intelligence*, vol.19, Oct.1982, pp.165-187.

[Kaplan 84]

S. Jerrold Kaplan: "Designing a Portable Natural Language Database Query System;" *ACM TODS*, vol.9 no.1, Mar.1984, pp.1-19.

[Keller 81a]

Arthur M. Keller and Gio Wiederhold: "Validation of Updates Against the Structural Database Model"; *Symposium on Reliability in Distributed Software and Database Systems*, July 1981, Pittsburgh PA, pp.195-199.

[Keller 81b]

A.M. Keller: *Updates to Relational Databases Through Views Involving Joins*; IBM Research Report, IBM San Jose and Stanford University, October 1981.

[Keller 82]

Arthur M. Keller: "Updates to Relational Databases through Views Involving Joins"; *Proc. 2nd International Conference on Databases: Improving Usability and Responsiveness*, Jerusalem, Israel, Academic Press, June 1982, pp.363-384.

[Keller 84a]

Arthur M. Keller and Marianne Winslett: "Approaches for Handling Incomplete Information and Nulls in Databases"; *IEEE Data Engineering Conference*, Los Angeles CA, April 1984.

[Keller 84b]

A.M. Keller and Jeffrey D. Ullman: "On Complementary and Independent Mappings on Databases"; *ACM-SIGMOD 1984 International Conference on Management of Data*, June 1984, Boston MA, publication, February 1986

[Keller 85a]

A.M. Keller: "Algorithms for Translating View Updates to Database Updates for Views Involving Selections, Projections, and Joins"; *ACM Principles of Database Systems*, March 1985, Portland OR, ACM SIGACT and SIGMOD.

[Keller 85b]

A.M. Keller and Marianne Winslett: "On the Use of an Extended Relational Model to Handle Changing Incomplete Information"; *IEEE Transactions on Software Engineering*, July 1985.

[Keller 85c]

Arthur M. Keller: *Updating Relational Databases Through Views*; Ph.D. dissertation, Stanford University, Computer Science Dept., report CS-85-1040, February 1985.

- [Keller 86a]
Arthur M. Keller: "Choosing a View Update Translator by Dialog at View Definition Time"; *IEEE Computer*, January 1986.
- [Keller 86b]
Arthur M. Keller: "Set-Theoretic Problems of Null Completion in Relational Databases"; *Information Processing Letters*, to appear in April 1986.
- [Keller 86c]
Arthur M. Keller: "Indexed File Access for Ada"; submitted for publication.
- [Keller 86d]
Arthur M. Keller: "On Bancilhon and Spyrtos' 'Update Semantics and Relational Views'"; revised and submitted for publication, January 1986.
- [Keller 86e]
Arthur M. Keller: "A Deadlock-Free Multi-Indexed B⁺-Tree" submitted for publication, February 1986.
- [Keller 86f]
Arthur M. Keller: "Updating Relational Databases Through Union Views"; submitted for publication, March 1986.
- [King 79]
Jonathan King: *Exploring the Use of Domain Knowledge for Query Processing Efficiency*; Stanford University Heuristic Programming Project paper HPP-79-30, Dec. 1979.
- [King 80a]
J. King: "Modelling Concepts for Reasoning about Access to Knowledge"; *Proceedings of the ACM Workshop on Data Abstraction, Data Bases, and Conceptual Modelling*, Pingree Park CO, June 23-26, 1980, *ACM-SIGPLAN Notices*, vol.16 no.1, Jan.1981.
- [King 80b]
J. King: "Intelligent Retrieval Planning"; *Proc. of the first National Conference on Artificial Intelligence*, Stanford CA, August 1980, pp.243-245.
- [King 81a]
Jonathan J. King: *Query Optimization by Semantic Reasoning*; PhD dissertation, Technical Report STAN-CS-81-857, Stanford University Computer Science Dept., May 1981.
- [King 81b]
Jonathan J. King: "QUIST: A System for Semantic Query Optimization in Relational Databases"; *VLDB 7*, Zaniolo and Delobel(eds), Sep.1981, pp.510-517.
- [Kuhn 81]
I.M. Kuhn and G. Wiederhold: "The Evolution of Ambulatory Medical Record Systems in the U.S."; *SCAMC 5*, IEEE, 1981, pp.80-85.
- [Kuhn 82]
Ingeborg M. Kuhn, Gio Wiederhold, Jonathan E. Rodnick, Diane Ramsey-Klee, Stanford Bennett, Donald D. Beck: *Automated Ambulatory Medical Record Systems in the U.S.*; Stanford CS Report STAN-CS-82-298, Stanford Univ., August 1982. Re-published as Chapter 14 in Bruce I. Blum (ed.): *Information Systems for Patient Care*, Springer Verlag 1984, pp.199-217.
- [Malachi 85]
Y. Malachi, Z. Manna, and R. Waldinger: "TABLOG: Functional and Relational Programming in One Framework"; *IEEE Software*, Jan.1986, pp.75-76.

[Milton 83]

J. Milton and G. Wiederhold: "Network and Relational Systems for VLSI Design: Contradictory Performance?"; *Database Engineering, Volume I*, Kim, Batory, Hevner, Katz, and Reiner(eds.), IEEE Computer Society Press, 1983, pp.167-170.

[Minoura 78]

T. Minoura: "Maximally Concurrent Transaction Processing"; Berkeley Workshop 3, 1978.

[Minoura 81]

Toshimi Minoura and Gio Wiederhold: "Resilient Extended True-Copy Token Scheme for a Distributed Database System"; *IEEE Symposium on Reliability in Distributed Software and Database Systems*, July 1981, Pittsburgh PA, pp.1-12.

[Minoura 82]

Toshimi Minoura and Gio Wiederhold: "Resilient Extended True-Copy Token Scheme for a Distributed Database System"; *IEEE Transactions on Software Engineering*, vol.SE-8 no.3, May 1982, pp.173-188.

[Minoura 83a]

Toshimi Minoura, Susan Owicki, and Gio Wiederhold: *True-Copy Token Scheme for a Distributed Database System*; Oregon State University Report 83-60-1, also being reviewed for a book chapter by Bhargava.

[Minoura 83b]

Toshimi Minoura, Susan Owicki, and Gio Wiederhold: *Consistent Distributed Database State Maintenance*; Oregon State University CS Report 83-60-2.

[Missikoff 84]

Michele Missikoff and Gio Wiederhold: "Towards a Unified Approach for Expert and Database Systems"; *Proceedings of the First Workshop on Expert Database Systems*, Kiawah Island, South Carolina, Oct.1984, Institute of Information Management, Technology and Policy, Univ. of South Carolina, vol.2, pp.186-206; also in *Expert Database Systems*, Larry Kerschberg (editor), Benjamin/Cummings, 1986, pages 383-399.

[Navathe 82]

Sham Navathe, Stefano Ceri, JinLi Dou, and Gio Wiederhold: "Vertical Partitioning for Physical and Distribution Design of Databases"; Stanford Computer Science report STAN-CS-82-957, Dec.1982. *ACM TODS*, vol.9 no.4, Dec.1984, pp.680-710.

[Olumi 83]

Mohamed Olumi, G. Wiederhold, C. Hauser, P. Lucas, and J. Mehl: "Software Project Databases"; *ACM-SIGMOD Proceedings*, May 1983, pp.124-134.

[Paige 82]

Robert Paige: "Applications of Finite Differencing to Database Integrity Control and Query/Transaction Optimization"; in *Bases Logiques pour Bases de Donnees*, Nicolas(Ed.), Onera-Cert, Toulouse, 1982.

[Paige 83a]

Robert Paige: "Transformational Programming — Applications to Algorithms and Systems"; *ACM-POPL '83*, 1983.

[Paige 83b]

Robert Paige: "Applications of Finite Differencing to Database Integrity Control and Query/Transaction Optimization"; *Advances in Database Theory*, vol.2, Minker, Nicolas, and Gallaire (eds.), 1983.

- [Pednault 84]
Edwin Pednault: "A New Approach to Classical Planning"; Artificial Intelligence Center, SRI International, submitted for publication, April 1984.
- [Qian 85a]
XiaoLei Qian: "FLASH Users Guide"; Second Version, Stanford University, CSD, KBMS project, April 1985.
- [Qian 85b]
XiaoLei Qian and Gio Wiederhold: "Data Definition Facility of CRITIAS"; *Proceedings of the 4th International Conference on Entity-Relationship Approach*, Oct.1985, pp.46-55.
- [Rathmann 84a]
Peter Rathmann: "Dynamic Data Structures on Optical Disks"; *IEEE Data Engineering Conf.*, Los Angeles CA, Apr.24-27, 1984.
- [Rathman 84b]
Peter Rathman: "A Tool for Optical Disk Data Structure Investigation"; Stanford University, Computer Science Dept., KBMS project, Spring 1984.
- [Rathmann 86]
Peter Rathmann: "The Cost of Locking"; submitted to *International Conference on Database Theory*, 1986.
- [Rowe 79a]
Neil Rowe: "Network Support for a Distributed Data Base System"; *Berkeley Workshop* 4, August 1979.
- [Rowe 81a]
Neil Rowe: "Rule-Based Statistical Calculations on a Database Abstract"; *1st LBL Workshop on Statistical Database Management*, Menlo Park CA, December 1981, pp.163-176.
- [Rowe 82a]
Neil Rowe: "Diophantine Compromise of a Statistical Database;"Chap. 3, Stanford Computer Science Dept. Report CS-82-948, 1982 and *Information Processing Letters*, June 1983.
- [Rowe 82b]
Neil Rowe: "Inheritance of Statistical Properties"; *AAAI-82 Conference*, Pittsburgh PA, August 18-20, 1982, pp.221-224.
- [Rowe 82c]
Neil Rowe: "On Some Arguable Claims in B. Shneiderman's Evaluation of Natural Language Interfaces to Databases"; *ACM-SIGMOD Record*, vol.13 no.1, Sep.1982.
- [Rowe 82d]
Neil Rowe: *Modelling Degrees of Item Interest for a General Database Query System*; Stanford Report STAN-CS-82-947, October 1982, 37 pp.
- [Rowe 82e]
Neil Rowe: "Inheritance of Statistical Properties"; *National Conference*, American Association for Artificial Intelligence, Pittsburgh PA, 1982, pp.221-314. Also part 2 of Stanford CS Report CS-82-948, December 1982.
- [Rowe 82f]
Neil Rowe: *Three Papers on Rule-based Estimation of Statistics on Databases*, Stanford Computer Science Report CS-82-948, Dec.1982.

- [Rowe 83a]
Neil Rowe: "Top-down Estimation of Statistics on a Database"; *Proceedings of the ACM-SIGMOD Conference*, May 1983, pp.135-145.
- [Rowe 83b]
Neil Rowe: "An Expert System for Statistical Estimates on Databases"; National Conference of the American Association for Artificial Intelligence, March 1983.
- [Rowe 83c]
Neil Rowe: "Some Experiments in Evaluation of an Expert System for Statistical Estimation on Databases"; *2nd Workshop on Statistical Database Management*, Lawrence Berkeley Lab technical report, September 1983.
- [Rowe 83d]
Neil Rowe: *Rule-Based Statistical Calculations on a Database Abstract*; Ph.D. dissertation, Stanford University Computer Science Dept. report STAN-CS-83-975, June 1983.
- [Rowe 84]
Neil Rowe: "Diophantine Inferences from Statistical Aggregates on Few-Valued Attributes"; *IEEE Data Engineering Conference*, Los Angeles CA, April 1984.
- [Sacca 83]
Domenico Sacca and Gio Wiederhold: "Partitioning in a Cluster of Processors"; *VLDB 9*, Florence Italy, October 31-November 2, 1983, extended abstract only. Also IBM Report RJ4076, full paper, 1983.
- [Sacca 83]
Domenico Sacca and Gio Wiederhold: "Partitioning in a Cluster of Processors"; *ACM TODS*, vol.10 no.1, Mar. 1985, pp.29-56.
- [Shaw 79]
David E. Shaw: *A Hierarchical Associative Architecture for the Parallel Evaluation of Relational Algebraic Database Primitives*; Stanford Computer Science Department Report CS-79-778, October 1979.
- [Shaw 80a]
D. Shaw: "A Relational Database Machine Architecture"; *Proceedings of the 1980 Workshop on Computer Architecture for Non-Numeric Processing*, Asilomar CA, Mar.1980, also *ACM-SIGMOD*, vol.X no.4, and *ACM-SIGIR*, vol.XV no.2, Apr.1980, pp.84-95.
- [Shaw 80b]
D. Shaw: *Knowledge-Based Retrieval on a Relational Databased Machine*; PhD dissertation, Stanford University, Computer Science Department report CS-80-823, Sep.1980.
- [Shaw 81]
D.E. Shaw, S.J. Stolfo, H. Ibrahim, B. Hillyer, G. Wiederhold and J.A. Andrews: "The NON-VON Database Machine: A Brief Overview"; *IEEE Database Engineering Bulletin*, vol.4 no.2, Dec.1981.
- [Shuey 86]
Richard Shuey and Gio Wiederhold: "Data Engineering and Information Systems"; *IEEE Computer Magazine*, vol.19 no.1, January 1986, pp.18-30.
- [Spooner 85]
David Spooner, Arthur.M. Keller, and Gio Wiederhold: *Framework for the Security Component of an Ada DBMS*; Tech. Report, IDA, December 1985. To appear in *VLDB 12*, Kyoto Aug. 1986.

- [SYTH 85]
ZaiSheng Song, JuanFen Yu, ShiWei Tang, and ChoChun Hsu: "Optical Disk FLASH"; Stanford University, Computer Science Dept., KBMS project, September 1985.
- [WBF 85]
M.G. Walker, R.L. Blum, and L.M. Fagan: "Minimycin: A Miniature Rule-Based System"; *M.D.Computing*, vol.2 no.4., p.21, 1985.
- [Walker 85]
M.G. Walker and R.L. Blum: "An Introduction to LISP"; *M.D.Computing*, vol.2 no.1., p.56, 1985.
- [Walker 86a]
M.G. Walker and R.L. Blum: *Towards Automated Discovery from Clinical Databases: the RADIX Project*; Stanford KSL Memo KSL-86-7, January 1986; submitted to Medinfo '86.
- [Walker 86b]
M.G. Walker: "On Inappropriate Responses and Reliability in Artificial Intelligence Systems"; In preparation.
- [Walker 86c]
M.G. Walker: "A Comparison of Indices for Detecting Clusters and Non-linearity in Scattergrams." In preparation.
- [Walker 86d]
M.G. Walker: "How Feasible is Automated Discovery?" In preparation for IEEE Expert.
- [Whang 81a]
K. Whang, G. Wiederhold, and D. Sagalowicz: "Separability: An Approach to Physical Database Design"; in *Proceedings of the Seventh International Conference on Very Large Data Bases*, Zaniolo and Delobel(eds.), Cannes, France, Sep.1981, pp.320-332.
- [Whang 81b]
K. Whang, G. Wiederhold, and D.Sagalowicz: *Separability as a Physical Database Design Methodology*; CSL TR-222 and STAN-CS-81-898, Stanford University, October, 1981.
- [Whang 82]
K.-Y. Whang, G. Wiederhold, and D. Sagalowicz: "Physical Design of Network Model Databases Using the Property of Separability"; in *Eighth International Conference on Very Large Data Bases*, McLeod and Villasenor (eds.), Mexico City, Sep.1982, pp.98-107.
- [Whang 83a]
K.-Y. Whang: "Physical Design Algorithms for Multiple Relational Databases"; submitted for publication, 1983.
- [Whang 83b]
K.-Y. Whang, G. Wiederhold, and D. Sagalowicz: "Estimating Block Accesses in Database Organizations, A Closed, Non-iterative Formula"; *Communications of the ACM*, vol.26 no.11, November 1983, pp.940-944.
- [Whang 84]
K.-Y. Whang, G. Wiederhold, and D. Sagalowicz: "Separability-An Approach to Physical Database Design"; *IEEE Transactions on Computers*, vol.c-33 no.3, March 1984, pp. 209-222.

[Whang 85]

K.-Y. Whang, G. Wiederhold, and D. Sagalowicz: "The Property of Separability and Its Application to Physical Database Design;" *Query Processing in Database Systems*, Kim, Batory, Reiner (eds.), Springer Verlag, 1985.

[Wiederhold 78a]

Gio Wiederhold: "Introducing Semantic Information into a Database Schema"; *Proceedings of the CIPS Session '78*, Canadian Information Processing Society, September, 1978, pp.338-391.

[Wiederhold 78b]

Gio Wiederhold: "Management of Semantic Information for Databases"; HPP-78-12, *Proceedings of the 3rd USA-Japan Conference*, Session 10-2-1, San Francisco, October 1978, pp.192-197.

[Wiederhold 79a]

Gio Wiederhold and Ramez ElMasri: *Structural Model for Database Systems*; Stanford University, Computer Science Department report CS-79-722, April 1979.

[Wiederhold 79b]

G. Wiederhold and R. ElMasri: "Errata for Database Design"; *Database Engineering Bulletin*, IEEE Technical Committee on Databases, 1979.

[Wiederhold 79c]

Gio Wiederhold and Ramez ElMasri: "The Structural Model for Database Design;" *Proceedings of the International Conference on Entity-Relationship Approach to Systems Analysis and Design*, North Holland, December 1979, pp.247-267.

[Wiederhold 79d]

Gio Wiederhold: "Databases for Economic Planning in India"; *Management Sciences and the Development of Asian Economies*, S. Torok (editor), Times Books International, Singapore, 1979.

[Wiederhold 80a]

Gio Wiederhold: *Databases in Health Care*; Stanford Computer Science Dept. Report 80-790, March 1980.

[Wiederhold 80b]

Gio Wiederhold, Anne Beetem, and Garrett Short: "A Database Approach to Communication in VLSI Design"; Technical report 196, Stanford University, Computer Systems Laboratory, 1980.

[Wiederhold 80c]

Gio Wiederhold: "New Technologies"; in *The Computer in the Doctor's Office*, Rienhoff and Abrams (eds.), North-Holland Publishing Company, IFIP, 1980, pp.263-264.

[Wiederhold 80d]

Gio Wiederhold (Translator R. Gritsch): *Datenbanken, Analyse—Design—Erfahrungen*; Band 1 Dateisysteme, R. Oldenbourg Verlag München 1980, Reihe Datenverarbeitung, 454 pp.

[Wiederhold 81a]

Gio Wiederhold: *Databases for Health Care*; Lecture Notes in Medical Informatics no.12, Lindberg and Reichertz(eds.), Springer-Verlag, Berlin, Heidelberg, New York, 1981, 75 pp.

[Wiederhold 81b]

Gio Wiederhold: *Binding in Information Processing*; Stanford University Computer Science Report STAN-CS-81-851, May 1981.

[Wiederhold 81c]

Gio Wiederhold: "Database Technology in Healthcare"; *Journal of Medical Systems*, Plenum, vol.5 no.3, Sept.1981, pp.175-196.

[Wiederhold 81d]

G. Wiederhold, J. Kaplan, and D. Sagalowicz: "Research in Knowledge Base Management Systems"; *ACM-SIGMOD Record*, vol.11 no.3, Apr.1981, pp.26-54.

[Wiederhold 82a]

Gio Wiederhold, J. Kaplan, and D. Sagalowicz: "Physical Database Design Research at Stanford"; *IEEE Database Engineering Bulletin*, vol.5 no.1, pp.39-41, March 1982, republished in *Database Engineering*, IEEE Computer Society, 1983.

[Wiederhold 82b]

Gio Wiederhold, Anne Beetem, and Garret Short: "A Database Approach to Communication in VLSI Design"; *IEEE Transactions on Design Automation*, vol.CAD-1 no.2, April 1982, pp.57-63; to be republished in *Digital VLSI Systems*, M.I. Elmasry (ed.), IEEE Press, 1985.

[Wiederhold 82c]

Gio Wiederhold: "Databases for Ambulatory Care"; *Proceedings of the Annual Conference of the American Medical Informatics Association*, May 1982, pp.79-85.

[Wiederhold 82d]

Gio Wiederhold: "A Method for the Design of Multi-Objective Databases"; *Proceedings of the Annual Meeting of the American Society of Mechanical Engineers*, June 1982, pp.161-165.

[Wiederhold 82e]

G. Wiederhold(ed.): *Second Symposium on Reliability in Distributed Software and Database Systems*; IEEE Pub.no.82CH1792-1, 1982.

[Wiederhold 82f]

Gio Wiederhold: "Applications of Computers in Medicine"; *Encyclopedia of Computer Science*, Ralston (ed.), Second edition; Van Nostrand-Reinhold 1982, pp.686-688, 934-940.

[Wiederhold 82g]

Gio Wiederhold: "Scientific Computing: A Scientific Group of Mixed Background Attempts to Cope with a Broad Spectrum of Problems": abstracted in *Roles of Industry and the University in Computer Research and Development*, National Academy Press, Washington DC, 1982, pp.60-66.

[Wiederhold 82h]

Gio Wiederhold: "Hospital Information Systems"; *Encyclopedia of Computer Science and Engineering*, 2nd edition, Ralston and Reilly (eds.), Van Nostrand Reinhold Company, 1982.

[Wiederhold 82i]

Gio Wiederhold: "Medical Applications"; *Encyclopedia of Computer Science and Engineering*, 2nd edition, Ralston and Reilly (eds.), Van Nostrand Reinhold Company, 1982.

[Wiederhold 82j]

Gio Wiederhold: "Databases and File-Systems"; in *The McGraw-Hill Handbook*, New York, Helms (ed.), 1982, pp.19-1-99.

[Wiederhold 83a]

Gio Wiederhold: *Database Design*; McGraw-Hill (in the Computer Science Series) Second edition, Jan.1983, 768 pp.

[Wiederhold 83b]

Gio Wiederhold: "Modeling Databases"; *Information Systems*, vol.29, 1983.

[Wiederhold 83c]

G. Wiederhold, J. Milton, and D. Sagalowicz: *The Knowledge-Based Management Systems project*; Stanford University, June 1983.

[Wiederhold 83d]

Gio Wiederhold: "Networking of Data and Information", *NCI Workshop on Role of Computers in Cancer Clinical Trials*, NIH, June 1983.

[Wiederhold 83e]

Gio Wiederhold: "Databases and Information Management"; *WIS Implementation Study Report*, Volume III - Background Information; Thomas H. Probert (ed.), IDA, Oct.1983, pp.75-137, 154.

[Wiederhold 83f]

Gio Wiederhold, Jack Milton, and Daniel Sagalowicz: "Applications of Artificial Intelligence in the Knowledge Based Management Systems Project"; *IEEE Database Engineering Bulletin*, vol.6 no.4, Dec.1983, pp.75-82; reprinted in *IEEE Database Engineering*, vol.2, 1984 Kim, Ries, Lochovsky (eds.), pp.257-264.

[Wiederhold 84a]

Gio Wiederhold: "Knowledge and Database Management"; *IEEE Software Premier Issue*, vol.1 no.1, January 1984, pp.63-73.

[Wiederhold 84b]

Gio Wiederhold: "Databases"; *IEEE Computer*, Centennial Issue, vol.17 no.10, October 1984, pp.211-223.

[Wiederhold 84c]

Gio Wiederhold: "Databases for Statistics"; in *Proceedings of the Data Base Management Conference*, Naval Post-Graduate School, Monterey CA, November 1984.

[Wiederhold 84d]

Gio Wiederhold: "A Metaphor for Distributed Databases"; report to VisiCorp, March 1984.

[Wiederhold 85a]

Gio Wiederhold, Robert L. Blum, and Michael Walker: "An Integration of Knowledge and Data Representation"; *Proceedings of the Islamorada Workshop*, Feb.1985, Computer Corporation of America, Cambridge MA; in *On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies* (Brodie, Mylopoulos, and Schmidt, eds.) Springer-Verlag, June 1986.

[Wiederhold 85b]

Gio Wiederhold and Paul D. Clayton: "Processing Biological Data in Real Time"; *M.D. Computing*, Springer Verlag, Vol.2 No.6, November 1985, pages 16-25.

[Wiederhold 85c]

Gio Wiederhold: "Knowledge Bases"; *Future Generations Computer Systems*, North-Holland, vol.1 no.4, May 1985, pp.223-235.

[Wiederhold 86a]

Gio Wiederhold: "Knowledge Versus Data"; Chapter 9 of *On Knowledge Base Management Systems: Integrating Artificial Intelligence and Database Technologies* (Brodie, Mylopoulos, and Schmidt, eds.) Springer-Verlag, Jun.1986.

[Wiederhold 86b]

Gio Wiederhold, Arthur M. Keller, Sham Navathe, David Spooner, Murray Berkowitz, Bill Brykczynski, and John Salasin: "Modularization of an Ada* Database System"; Stanford, February 1986; extended abstract in Proc. 6th Advanced Database Symposium, Information Processing Society of Japan, August 1986; to be presented in Beijing, Aug.1986.

[Wiederhold 86c]

Gio Wiederhold: "Views, Objects, and Databases"; Stanford, March 1986, Proc. of the Workshop on Integrated Manufacturing Systems, Naval Post-graduate School, Monterey CA, 14 Apr.1986.

[Wiederhold 86d]

Gio Wiederhold John Smith, and Phillip Kaufman: "Models for Engineering Information Systems"; abstract in *Proceedings of the 1985 VHSIC Conference*, Silver Spring, Maryland, December 1985.

[Wiederhold 86e]

Gio Wiederhold and XiaoLei Qian: "A New Relationship Type: The Identity Connection"; Stanford March 1986, submitted to the *5th International Conference on Entity-Relationship Approach*, Nov.1986.

[Wiederhold 86f]

Gio Wiederhold: "Standardization and Artificial Intelligence"; *ASTM Standardization News*, vol.14 no.4, Apr.1986, pp.10-11.

[WiederholdV 84]

Voy Wiederhold, Lei-Hou Chee, James Theodore, Edward Stinson, and Gio Wiederhold: "A Comparison of Two Database Systems for Heart-Lung Transplant Data"; *Computers in Cardiology*, IEEE, September 18-21, 1984, Salt Lake City UT.

[Winslett 84]

Marianne Winslett and Gio Wiederhold: "Relational and Entity-Relationship Model Databases in VLSI Design"; *IEEE Database Engineering Bulletin*, vol.7 no.2, Jun.1984, pp.61-66.

[Winslett 85a]

Marianne Winslett, Richard Berlin, Thomas Payne, and Gio Wiederhold: "Relational and Entity-Relationship Model Databases and Specialized Design Files in VLSI Design"; *Proceedings of the ACM Design Automation Conference*, Jun.1985, pp.410-416.

[Winslett 85b]

Marianne Winslett: "A Model-Theoretic Approach to Updating Logical Databases"; Stanford Computer Science Dept. Technical Report, Jan. 1986.

[Winslett 86]

Marianne Winslett: "A Model-Theoretic Approach to Updating Logical Databases (Extended Abstract)"; *ACM Conference on Principles of Database Systems*, Mar. 1986.

11. Related Reports and Papers

[Bachman 72]

C.W. Bachman: "The Evolution of Storage Structures"; *CACM*, vol.15 no.7, July 1972, pp.628-634.

[Bancilhon 81]

F.B. Bancilhon and N. Spryatos: "Update Semantics of Relational Views" *ACM TODS*, vol.6 no.4, Dec.1981, pp.557-575.

[Barbera 82]

D. Barbera and H. Garcia-Molina: "How Expensive is Data Replication"; *Proceedings of the Third International Conference on Distributed Computing Systems*, pp.263-268, Oct.1982.

[Barr 81]

A. Barr, E. Feigenbaum, and P. Cohen (eds.): *The Handbook of Artificial Intelligence*, vol. I; Kaufman Publishers, Los Altos CA, 1981

[Belady 77]

L.A. Belady, P.M. Merlin: "Evolving Parts and Relations - A Model of System Families"; *IBM Research Report RC6677*, Aug.1977.

[Bennett 82]

Jack Bennett: "A Database Management System for Design Engineers"; *ACM, IEEE Nineteenth Design Automation Conference Proceedings*, June 14-16, 1982, pp.268-273.

[Blasgen 77]

M. Blasgen and K. Eswaren: "Storage and Access in Relational Data Bases"; *IBM Systems Journal*, vol.16 no.4, 1977.

[Bleier 67]

R.E. Bleier: "Treating Hierarchical Data Structures in the SDC Time-shared Data Management System (TDMS)"; *Proceedings of the 22nd ACM National Conference 1967*, MDI Publishers, Wayne PA, pp.41-49.

[Blum 83]

Robert L. Blum: "Clinical Decision Making Aboard the Starship Enterprise"; *SCAMC 83*, IEEE, April 1983.

[Blum 86a]

R.L. Blum: "Computer-Assisted Design of Studies Using Routine Clinical Data: Prednisone Elevates Cholesterol"; Tentatively accepted for publication in *Annals of Internal Medicine*. Currently being revised for publication.

[Blum 86b]

R.L. Blum: "Two-Stage Regression: Application to a Time-Oriented Clinical Database." Submitted for publication.

[Boyce 75]

R. Boyce, D.Chamberlin, M. Hammer and W. King: "Specifying Queries as Relational Expressions: The SQUARE Data Sublanguage"; *CACM*, vol.18 no.11, Nov.1975, pp.621-628.

[Brachman 83]

Ronald J. Brachman, R.E. Fikes, and H.J. Levesque: "KRYPTON: A Functional Approach to Knowledge Representation"; *IEEE Computer*, vol.16 no.10, Oct.1983, pp.67-73.

- [Burkhard 76]
W.A. Burkhard: "Hashing and Trie Algorithms for Partial Match Retrieval"; *ACM TODS*, vol.1 no.2, June 1976, pp.175-187.
- [Ceri 84]
S. Ceri and G. Pelagatti: *Distributed Databases: Principles and Systems*; McGraw-Hill, 1984.
- [Ceri 86]
S. Ceri and G. Gottlob: "Normalization of Relations and Prolog"; submitted for publication, 1986.
- [Chamberlin 74]
D.D. Chamberlin and R.F. Boyce: "SEQUEL: A Structured English Query Language"; *Proceedings of the ACM-SIGMOD Workshop on Data Description, Access and Control*, May 1974, pp.249-264.
- [Chang 76]
C.L. Chang: "DEDUCE: A Deductive Query Language for Relational Data Bases"; in: *Pattern Recognition and Artificial Intelligence*, Academic Press, 1976, pp.108-134.
- [Chang 76]
S. Chang, M. O'Brien, J. Read, A. Borovec, W. Cheng, and J.S. Ke: "Design Considerations of a Data Base System in a Clinical Network Environment"; *Proceedings of the 1976 NCC*, AFIPS Press, Arlington VA, 1976, vol.45, pp.277-286.
- [Chen 77]
P.P. Chen: "The Entity-Relationship Model - A Basis for the Enterprise View of Data" *Proceedings of the 1977 NCC*, AFIPS Press, Arlington VA, 1977, vol.46, pp.77-84.
- [Cherubini 84]
R. Cherubini: "X9: A System for Creating and Delivering On-line Documentation"; Internal Memorandum, Digital Equipment Corporation, 1984.
- [Cline 85]
T. Cline, W. Fong, M.G. Walker, and S. Rosenberg.: "Photolithography Advisor;" *SIGART Newsletter*, April 1985, no.92, p.42, Special Section on AI in Engineering, Sriram and Joobani(eds.).
- [CODASYL 71]
Data Base Task Group of CODASYL Programming Language Committee; ACM report, April 1971
- [CODASYL 74]
CODASYL Data Description Language, Journal of Development June 73; NBS Handbook 113, Government Printing Office, Washington DC, Jan.1974, 155pp.
- [Codd 70]
E.F. Codd: "A Relational Model of Data for Large Shared Data Banks"; *CACM*, vol.13 no.6, June 1970, pp.377-387.
- [Codd 71]
E.F. Codd: "A Data Base Sublanguage Founded on the Relational Calculus"; *Proceedings of the ACM SIGFIDET Workshop on Data Description, Access and Control*, San Diego, Nov.1971, pp.35-68.

- [Codd 74]
E.F. Codd: "Seven Steps to Rendezvous with the Casual User"; in: *Data Base Management*, Klimbie and Koffeman(eds.), North Holland, 1974, pp.179-200.
- [Cohen 74]
S.N. Cohen et al: "Computer Monitoring and Reporting of Drug Interactions"; *Proceedings of MEDINFO 1974*, IFIP Congress, Anderson and Forsythe, (eds.), North-Holland, 1974.
- [Duda 78]
R.O. Duda, P.E. Hart, P. Barrett, J.G. Gaschnig, K. Konolidge, R. Reboh, and J. Slocum: *Development of the Prospector Consultation System for Mineral Exploration*; Final Report, SRI project 5821 and 6415, Oct.1978.
- [ElMasri 84]
Ramez ElMasri and Shamkant Navathe: "Object Integration in Database Design"; *IEEE Data Engineering Conference*, Los Angeles, Apr.24-27, 1984.
- [Everest 74]
Gordon C. Everest: "The Objectives of Data Base Management"; *Proceedings of the 4th International Symposium on Computer and Information Sciences*, Plenum Press, 1974, pp.1-35.
- [Finkelstein 82]
Sheldon J. Finkelstein, Mario Schkolnick, and Paul Tiberio: *A Physical Database Design Tool for Relational Databases*; IBM Research Report, 1982.
- [FCWR 85]
W. Fong, T. Cline, M.G. Walker, and S. Rosenberg; "An Expert System for Photolithography"; Semicon East, October 1985.
- [Fries 75]
James F. Fries, Alison Harlow, and Gio Wiederhold: "Experience-based Computer Consultation"; *IEEE Cybernetic Systems*, Sept. 1975.
- [Fries 79]
James F. Fries and Dennis McShane: "ARAMIS, A National Chronic Disease Databank System"; *Proceedings of the 3rd Symposium on Computer Applications in Medical Care*, Oct.1979, Washington DC, IEEE, pp.798-801.
- [Garcia 77]
H. Garcia-Molina: "Overview and Bibliography of Distributed Databases"; Computer Science Department, Stanford University.
- [GarciaMolina 81]
H. Garcia-Molina: *Using Semantic Knowledge For Transaction Processing in Distributed Databases*; Technical Report 285, Dept. of Electrical Engineering and Computer Science, Princeton University, April 1981.
- [Germano 80]
F. Germano: *Automatic Transaction Decomposition in a Distributed CODASYL Prototype System*; Ph.D. Thesis, Wharton School, University of Pennsylvania, 1980.
- [Ghosh 75]
S.P. Ghosh: "Consecutive Storage of Relevant Records with Redundancy"; *CACM*, vol.16 no.8, August 1975, pp.464-471.

- [Gilbert 82]
E.J. Gilbert: *Algorithm Partitioning Tools for a High-Performance Multiprocessor*; PhD. Thesis, Stanford University, Dec.1982.
- [Grosz 77a]
B.J. Grosz: *The Representation and Use of Focus in Dialog Understanding*; PhD. Thesis, The University of California at Berkeley, June 1977.
- [Grosz 77b]
B.J. Grosz: "The Representation and Use of Focus in a System for Understanding Dialogs"; *Proceedings of the 5th IJCAI*, Cambridge MA, 1977, vol.1, pp.67-76.
- [Hammer 76]
M.M. Hammer and A .Chan: "Index Selection in a Self-Adaptive Database Management System"; *Proceedings of the ACM-SIGMOD Conference*, Washington DC, June 1976, pp.1-8.
- [Hammer 77]
M.M. Hammer: "Self-Adaptive Automatic Data Base Design"; *Proc AFIPS 1977 NCC*, pp.123-129.
- [Hammer 78]
M.M. Hammer and D. McLeod: "The Semantic Data Model, a Modelling Mechanism for Database Applications"; *ACM-SIGMOD International Conference on Management of Data*, Austin TX, 1978, pp.26-36.
- [Hammer 80]
M.M. Hammer and D. Shipman: "Reliability Mechanisms for SDD-1"; *ACM TODS*, vol.5 no.4, Dec.1980, pp.431-466.
- [Harris 77]
L.R. Harris: "ROBOT: A High Performance Natural Language Processor for Data Base Query"; *ACM-SIGART Newsletter*, no.61, Feb.1977, pp.39-40.
- [Hayes 85]
Barabara Hayes-Roth: "BB1: An Architecture for Blackboard Systems that Control, Explain, and Learn About Their Own Behaviour"; Stanford University Report STAN-CS-84-1034, December 1984; *AI Journal*, 1985.
- [Held 76]
G.D. Held, M.R. Stonebraker and E. Wong: "INGRES: A Relational Data Base System"; *AFIPS Conference Proceedings*, vol.44, 1975, pp.416.
- [Hendrix 75]
G.G. Hendrix: *Expanding the Utility of Semantic Networks through Partitioning*; SRI Artificial Intelligence Group Technical Note 105, June 1975.
- [Hendrix 77]
G.G. Hendrix: "Human Engineering for Applied Natural Language Processing"; *Proceedings of the 5th IJCAI*, Cambridge MA, 1977, vol.1, pp.183-191.
- [Hendrix 80]
G.G. Hendrix: "Mediating the Views of Databases and Database Users"; ACM Pingree Park Workshop, Brodie(ed.), Jun.1980, pp.131-132.
- [Hodges 83]
A. Hodges: *Alan Turing: The Enigma*; p.361, Simon and Schuster, 1983.

[Horvitz 86]

E.J. Horvitz and D.E. Heckerman; "Modular belief updates and the inconsistent use of measures of certainty in artificial intelligence research"; (Memo KSL-85-57); to appear in *Uncertainty in Artificial Intelligence*, Amsterdam: North-Holland, 1986.

[Kaplan 79]

S.J. Kaplan: *Cooperative Responses from a Portable Natural Language Data Base Query System*; PhD. Dissertation, Dept. of Computer and Information Science, University of Pennsylvania, Philadelphia PA, 1979.

[Katz 84]

R.H. Katz and T.J. Lehman: "Database Support for Versions and Alternatives of Large Design Files"; *IEEE Transactions on Software Engineering*, vol.SE-10 no.2, Mar.1984.

[Knuth 73]

D.E. Knuth: *The Art of Computer Programming*; vol.3: *Sorting and Searching*; Addison-Wesley, 1973.

[LaFue 82]

Gilles M.E. LaFue: "Semantic Integrity Dependencies and Delayed Integrity Checking"; *Very Large Data Bases 8*, McLeod and Villasenor(eds.) 1982, pp.292-299.

[LaFue 83]

Gilles M.E. LaFue: "Basic Decisions about Linking an Expert System with a DBMS: A Case Study"; *IEEE Database Engineering Bulletin*, vol.6 no.4, Dec.1983, pp.56-64; reprinted in *IEEE Database Engineering*, vol.2, 1984 Kim, Ries, Lochovsky(eds.), pp.238-246.

[LaFue 85]

G.M.E. LaFue and R. Smith.: "Implementation of A Semantic Integrity Manager With a Knowledge Representation System "; *EDBS 1*, vol.1, Oct.1983, pp.172-185.

[Lee 78]

R.M. Lee: "Conversational Aspects of Database Interactions"; *Conference on Very Large Data Bases*, Berlin, Germany, 1978.

[Lehnert 77]

W. Lehnert: "Human and Computational Question Answering"; *Cognitive Science* vol.1 no.1, 1977.

[Lucas 75]

H.C. Lucas: "Performance and Use of an Information System"; *Management Science*, vol.21 no.8, April 1975, pp.908-919.

[Mackinlay 83]

J. Mackinlay, "Intelligent Presentation: The Generation Problem for User Interfaces," Stanford University Heuristic Programming Project Tech. Report, March 1983.

[Mackinlay 86]

J. Mackinlay, "Using Composition to Design Presentations Automatically," Stanford University Computer Science Dept. Tech. Report, March 1986.

[Marill 75]

T. Marill and D. Stern: "The Datacomputer, A Network Data Utility"; *Proceedings of the 1975 NCC, AFIPS*, vol.44, pp.389-395.

- [Martin 77]
N. Martin, P. Friedland, J. King, and M.J. Stefik: "Knowledge Base Management for Experiment Planning"; Stanford University, Heuristic Programming Project Paper HPP-77-19 Report, August 1977.
- [McDermott 75]
D.V. McDermott: *Very Large Planner-type Data Bases*; M.I.T. Artificial Intelligence Laboratory Memo 339, September 1975.
- [Montgomery 72]
C.A. Montgomery: "Is Natural Language an Unnatural Query Language"; *Proceedings of the 27th Conference, ACM*, 1972, pp.1075-1078.
- [Moore 78]
R.C. Moore: *Handling Complex Queries in a Distributed Data Base*; Technical Report, Artificial Intelligence Center, SRI International, 1978.
- [Mylopoulos 75]
J. Mylopoulos and N. Roussopoulos: "Using Semantic Networks for Data Base Management"; *Conference on Very Large Data Bases*, Framingham MA, Sept.1975, pp.144- 172.
- [Mylopoulos 85]
J. Mylopoulos, A. Borgida., S. Greenspan, and H.K.T. Wong: "Information System Design at the Conceptual Level - The TAXIS Project"; *IEEE Database Engineering Bulletin*, vol.7 no.4, Dec.1984, pp.4-9.
- [Navathe 82]
Shamkant B. Navathe and S.G. Gadgil: "A Methodology for View Integration in Logical Database Design"; *Conference on Very Large Data Bases 8*, McLeod and Villasenor(eds.), Mexico City, Sep.1982, pp.142-164.
- [Nava 84b]
S.B. Navathe, T. Sashidhar, and R. ElMasri: "Relationship Merging in Schema Integration"; *Proceedings of the 10th VLDB Conference*, Singapore, Aug.1984.
- [Novak 76]
G.S. Novak: "Computer Understanding of Physics Problems Stated in Natural Language"; *American Journal of Computational Linguistics*, Microfiche 53, 1976.
- [Parnas 76]
David L. Parnas: "On the Design and Development of Program Families"; *IEEE Transactions on Software Engineering*, vol.SE-2 no.1, March 1976.
- [Patel-Schneider 84]
Peter F. Patel-Schneider, Hector J. Levesque, and Ronald J. Brachman: *ARGON, Knowledge Management meets Information Retrieval*; Technical Report, Fairchild Laboratory for AI Research, Palo Alto CA, received May 1984.
- [Petrick 76]
S.R. Petrick: "On Natural Language Based Computer Systems"; *IBM Journal of Research and Development*, vol.20 no.4, July 1976.
- [Reddy 76]
Raj Reddy, L. Erman, R. Fennel, and R. Neely: "The HEARSAY Speech Understanding System: An Example of the Recognition Process"; *IEEE Transactions on Computers*, vol.C-25, pp.427-431.

- [Rivest 76]
R.L. Rivest: "On Self-organizing Sequential Search Heuristics"; *CACM*, vol.19 no.2, February 1976.
- [Rochkind 75]
Marc J. Rochkind: "The Source Code Control System", *IEEE Transactions on Software Engineering*, vol.SE-1 no.4, Dec.1975.
- [Roussopoulos 77]
N. Roussopoulos: *A Semantic Network Model of Databases*; PhD. Thesis, Computer Science Department, University of Toronto, 1977.
- [Rowe 76]
Neil Rowe: "Grammar as a Programming Language"; *Creative Computing*, January-February 1978, pp.80-86.
- [Rowe 79b]
Neil Rowe: "Sine POLYS: Some Geometrical Explorations"; *Creative Computing*, December 1979, pp.92-95.
- [Rowe 80a]
Neil Rowe: "Inductive Common Ground Tutoring"; *Computers and Education*, vol.4 no.2, 1980, pp.177-187.
- [Rowe 80b]
Neil Rowe: "Property List Structures"; *Creative Computing*, October 1980, pp.126-130.
- [Rowe 81b]
Neil Rowe: "Some Rules for Good Simulations"; *Educational Computer*, November-December 1981, pp.37-40.
- [Rowe 82g]
Neil C. Rowe: "More POLYS"; *Creative Computing*, April 1982.
- [Sacerdoti 77a]
E.D. Sacerdoti: "Language Access to Distributed Data With Error Recovery"; *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, Cambridge MA, Aug.1977.
- [Sacerdoti 77b]
E.D. Sacerdoti: *A Structure for Plans and Behavior*; Elsevier North-Holland, 1977.
- [Sagalowicz 77]
D. Sagalowicz: "IDA: An Intelligent Data Access Program"; *Conference on Very Large Data Bases 3*, Tokyo, Japan, Oct. 1977.
- [Schkolnick 77]
M. Schkolnick: "A Clustering Algorithm for Hierarchical Structures"; *ACM TODS*, vol.2 no.1, March 1977, pp.27-44.
- [Selinger 79]
P. Selinger et al: *Access Path Selection in a Relational Database Management System*; IBM Technical Report RJ2429, 1979.
- [Shipman 81]
D.W. Shipman: "The Functional Data Model and the Data Language DAPLEX"; *ACM TODS*, vol.6 no.1, Mar.1981, pp.140-173.

[Shneiderman 73]

B. Shneiderman: "Optimum Data Base Reorganization Points"; *CACM*, vol.16 no.6, June 1973, pp.362-365

[Shortliffe 73]

E. Shortliffe, S.G. Axline, B.G. Buchanan, T.C. Merigan and S.N. Cohen: "An Artificial Intelligence Program to Advise Physicians Regarding Antimicrobial Therapy"; *Computers and Biomedical Research*, vol.6 no.6, Dec.1973, pp.544-560.

[Shortliffe 76]

E.H. Shortliffe: *Computer-Based Medical Consultations: MYCIN*; American Elsevier, 1976.

[Shortliffe 79]

E.H. Shortliffe, B.G. Buchanan, and E.A. Feigenbaum: "Knowledge Engineering for Medical Decision Making: A Review of Computer-Based Decision Aids"; *Proceedings of the IEEE*, vol.67 no.9, 1979, pp.1207-1223.

[Sibley 76]

E.H. Sibley and J.P. Fry: "Evolution of Data Base Management Systems"; *ACM Computing Surveys*, vol.8 no.1, March 1976, pp.7-42

[Sowa 76]

J.F. Sowa: "Conceptual Graphs for a Data Base Interface"; *IBM Journal of Research and Development*, vol.20 no.4, July 1976, pp.336-357.

[Steel 74]

T.B. Steel, Jr.: "Data Base Systems—Implications for Commerce and Industry"; in: *Data Base Management Systems*, Jardine, ed., North Holland 1974.

[Steel 75]

T.B. Steel, Jr.: *ANSI/X3/SPARC Study Group on Data Base Management Systems, Interim Report 75-02-08*; FDT (ACM-SIGMOD), vol.7 no.2, 1975.

[Stocker 77]

P.M. Stocker: "The Structuring of Data Bases at the Implementation Level"; in *Architecture and Models in Data Base Management Systems*, Nijssen, (ed.), North-Holland, 1977, pp.261-276.

[Taylor 74]

R.W. Taylor: "When Are Pointer Arrays Better Than Chains"; *Proceedings of the 1974 National Conference*, Nov.1974.

[Thompson 75]

F.B. Thompson and B.H. Thompson: "Practical Natural Language Processing: the REL System as Prototype"; in: *Advances in Computers 13*, Rubinoff and Yovits, (eds.), Academic Press, 1975, pp.109-168.

[Tuel 78]

W.G. Tuel: "Reorganization Points for Linearly Growing Files"; *ACM TODS*, vol.3 no.1, Sept.1978, pp.32-40.

[Walker 77]

D.E. Walker et al.: "An Overview of Speech Understanding Research at SRI"; *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, Cambridge MA, Aug.1977.

[Waltz 75]

D.L. Waltz: "Natural Language Access to a Large Database: An Engineering Approach"; *Proceedings of the 4th International Joint Conference on Artificial Intelligence*, Tbilisi USSR, Sept.1975, pp.868-872.

[Warren 81]

David H.D. Warren: "Efficient Processing of Interactive Relational Databases Queries Expressed in Logic"; *VLDB 7*, Zaniola and Delobel (eds.), Cannes, France, Sep.1981, pp.272-281.

[Warren 82]

David H.D. Warren and Fernando C.N. Pereira: "An Efficient Easily Adaptable System for Interpreting Natural Language Queries"; *American Journal of Computational Linguistics*, vol.8 no.3-4, July-Dec.1982, pp.110-122.

[Webber 84]

Bonnie Webber and T. Finin: "In Response: Next Steps in Natural Language Interaction;" *Artificial Intelligence Applications for Business*, Reitman (ed.), Ablex 1984, pp.211-234.

[Weyl 75]

S. Weyl, J. Fries, G. Wiederhold, and F. Germano: "A Modular Self-Describing Databank System;" *Computers and Biomedical Research*, vol.8, 1975, pp.279-293.

[Wiederhold 75]

G. Wiederhold, J.F. Fries and S. Weyl: "Structured Organization of Clinical Data Bases"; *Proceedings of the 1975 NCC*, AFIPS vol.44, pp.479-486.

[Wiederhold 77]

Gio Wiederhold: *Database Design*; McGraw-Hill, 1977, Chapter 7.

[Woods 73]

W.A. Woods: "Progress in Natural Language Understanding, An Application to Lunar Geology"; *Proceedings of the 1973 NCC*, AFIPS, vol.42, pp.441-450.

[Yao 76]

S.B. Yao, K.S. Das, and T.J. Teory: "A Dynamic Database Reorganization Algorithm;" *ACM TODS*, vol.1 no.2, June 1976, pp.159-174.

[Yao 78]

S.B. Yao and D. DeJong: "Evaluation of Database Access Paths"; *Proceedings of the ACM-SIGMOD*, Austin TX, May-June 1978, pp.66-77.

[Yao 79]

S.B. Yao: "Optimization of Query Evaluation Algorithms"; *ACM TODS*, vol.4 no.2, Jun.1979, pp.133-155.

[Zloof 75]

M. Zloof: "Query by Example"; *Proceedings of the 1975 NCC*, AFIPS vol.44, 1975, pp.431-438.

STRUCTURAL vs. APPLICATION KNOWLEDGE FOR IMPROVED DATABASE INTERFACES

Gio Wiederhold

Associate Professor (Research)

Departments of Medicine and Computer Science

KBMS PROJECT

Computer Science Department

Stanford University

Stanford, California



OUTLINE

Definitions

Objectives

Types of Knowledge

Examples of Use of Knowledge

Conclusion

Our View of the Data versus Knowledge

DATA represents FACTS

DATA are Objectively Verifiable
Mechanically Obtainable

KNOWLEDGE

represents ABSTRACTION

KNOWLEDGE is based on
EDUCATION
EXPERIENCE

Knowledge is MORE GENERAL than Data

Knowledge is LESS PRECISE than Data

Test:

If you'd let a clerk update it : DATA

If you'd trust an expert only : KNOWLEDGE

Examples

- (1) Mr. Chu's age is 43 → data
- (2) Middle-age is 35 to 50 → knowledge
- (3) People of middle-age are careful → knowledge
- (4) Mr. Chu is careful → ?

•Deduced from (1,2,3) → knowledge

•Observed independently → data
Can be used to strengthen (3)

Objectives

Applying Knowledge to Databases

Manage facts needed to operate enterprises

Manage voluminous facts efficiently

Share facts to make consistent decisions

Reduce large quantities of facts to information

Be responsive to user demands

How?

manage higher level abstractions

Make such information accessible:

In form — without intermediaries

In time — without delay

Problems of Databases

Addressed by
Knowledge-based Approaches

- User interface queries
responses
 - Provide linkages to analytic and
deductive inference procedures
 - Reduce facts to higher level concepts
 - Avoid delay due to large data quantities
 - Avoid delay due to wide distribution of data
-

Example why queries are not well handled in databases

•Query Formulation by statements

Retrieve name from employee
where age > 50
and salary < average (retrieve
salary from employee
where department = 'sales')

→ Large probability of errors by casual users:
Did the user intend to find any employee,
or only sales employees?

•Query Response (for above)

→ Nil !

What does that mean:

- 1 No employees older than 50 years old
- 2 No employees in sales
- 3 No low paid employees over 50 years old.

Categorization of Knowledge

- | | |
|-----------------------------------|-----------------------------|
| A. Application specific knowledge | (program specific) |
| B. Focus management k. | (application type specific) |
| C. General procedural k. | (program type specific) |
| D. Structural k. | (database specific) |
| E. Derived k. | (content specific) |
| F. Data domain k. | (compiler specific) |
| G. System control k. | (input/output specific) |

Here we will consider only
Knowledge at the higher levels

D. Structural Knowledge

Knowledge which is obtained when
a database is designed:

Components

- Entities modeled by relations
 - Relations as in the relational model.
- Relationships modeled by connections
 - Connections specify constraints of tuples in connected relations.

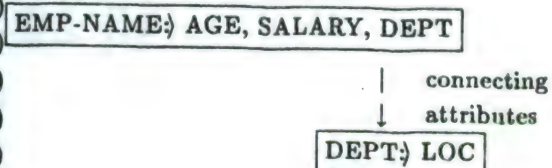
Formalization permits transformations
which are
◦provably correct
◦applicable to many instances

4 Connection Types

Ownership	1:N
Reference	M:1
Subset	1:1 partial
Identity	1:1 asynchronous

(M:N is constructed out of 2 connections and 1 relation
so that there are 4 distinct semantic types.)

Structural Connections



Two tuples are connected
if they have the same value for
the connecting attributes.

Connections specify constraints

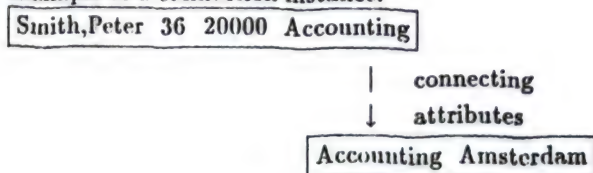
Disadvantage of constraints:
can not place in files anything
Junk — Valid exceptions

Benefits of constraints:

- A: prevents errors
- B: exceptions hard to process
- C: lack of exceptions enables optimization

More on connections

Example of a connection instance:



Connections are described by the quadruple:

{C-NAME: C-TYPE, SOURCE → DESTINATION, EXTRA-CONSTRAINTS}.

C-NAME identifies the relationship

C-TYPE designates one of the four types SOURCE and

DESTINATION identify the two relations and their attributes.

EXTRA-CONSTRAINTS attaches quantitative limits, such as $m < 40$ for the student tuples per class.

Ownership Connection: 1:N

Semantics:

Describe dependent tuples (ex. SKILL)

a generalization of dependent attributes (ex. AGE)

EMP-NAME : AGE, SAL, DEPT

Owner relation

|
*

EMP-NAME, SKILL : QUALIFICATION

Nest Relation

Static constraint:

• Every tuple in nest must have owner.

Dynamic constraint consequence:

- Cannot insert unowned tuples in nest
- Deletion of owner tuples causes deletion of all owned tuples.

Reference Connection: M:1

Semantics:

Describe abstract attributes in more detail

Referenced Relation

DEPT : LOC

↑

EMP-NAME : AGE, SAL, DEPT

Referencing Relation

Static constraint:

- Every tuple in referencing relation must be connected to a tuple in referenced relation.

Dynamic constraint consequence:

- Cannot insert referencing tuples if reference is not there
- Deletion of referenced tuples is prohibited.

Subset Connection 1:1 partial

Semantics:

Define a subset of tuples from a base relation that have additional properties:

EMP-NAME : AGE, SAL, JOB

Base relation

|
U

EMP-NAME : PLANT

Subrelation

Two variants

1. Restriction – membership in subrelation determined by data values.
2. Non-Restriction – membership determined externally.

Static constraint:

- Every tuple in subrelation must be connected to a tuple in base relation.

Dynamic constraint consequence:

- Cannot insert sub tuples if base tuple is not there
- Deletion of base tuple forces deletion of sub tuple.

Identity Connection 1:1 asynchronous

Semantics:

Replicated data to be made *eventually* identical.

Applies especially to autonomous distributed computers.

As long as distributed systems attempt to behave as if they were centralized systems, (concurrency control, recovery) conceptual modeling of distribution was not needed.

EMP-NAME :} HOURS-WORKED

Primary relation

EMP-NAME :} HOURS-WORKED

Secondary relation

Dynamic constraint:

only

• Eventually the values must be identical.

◦ Time constraints

1. a time limit: less than $\Delta = 1$ hour,

2. a time point: @5pm

3. an event trigger: prior !pay

where pay names the write-pay-check transaction.

more on identity connection

◦ attributes

always the key attributes

but all attributes may participate.

Derived data can participate

Examples:

1. In a manufacturing environment:

in the factory the inventory increases every minute
information has to be available to sales only daily

factory-stock: identity,

factory-inv. {p-no, made} —

sales-inv. {p-id, avail}, @7am;

2. The hours worked are needed weekly for paychecks

pay-hours: identity,

work. {ssn, SUM(hours | group by ssn) —

pay. {ssn, payhours}, @Fri.5pm;

3. Bank accounts are at the local branch when active

updated centrally at night with interbank transactions

accounts: identity,

branch. {ac-no, amount} —

central. {ac-no, amount}, !any(tx);

Effect

These 4 Connections embody much knowledge about the database.

This knowledge is needed by

a user or

a programmer

to formulate queries and updates correctly

but can also be used to automate part of the database interface

using the Reference Constraint

employee → department

"All employees must belong to a department"

Effect

1. Cannot delete a department record while there are still employees

2. Cannot enter an employee record without proper value in department field

1. All employees in all departments = all employees

2. All employees in one department are disjoint from any employee in other department

i.e.: If payrolls are processed department by department, then each employee gets one, and only one, paycheck.

Result: Less decision making and fewer programmer errors

Automatic Summarization

Problem: users cannot be expected to know the content of the database

• Can easily retrieve too much:

— Reference Knowledge is used

Trigger: More output than expected (if VDT, > 1 screen)

→ present abstractions

Look for common referenced classes

Output class descriptions, not detail

Example:

Show SHIPS in ROTTERDAM on 13 DEC 84

Response for 44 SHIPS

40 SHIPS COUNTRY= NETHERLANDS

1 ROYAL WARRIOR

1 IMANU MARU

1 GOLDER BEAR

1 NORSK FLOT 1

Summarization

Selects References in order (1,2,3, < 6)

Selects Best Clustering

To reduce entries to screen size (say =22)

if none goes to next reference

If no more references - gives up

User can continue with

an elliptical query

and Dutch SHIPS ?

Next Summarization by class

30 Ships Class = Freighter

9 Ships Class = Tanker

1 SS Volendam

1 Hoek Ferry

1 De Gezonde Apotheek

1 Royal Warrior

1 Tugger Tough

Next elliptical query is either

and Freighters

and Tankers

using Ownership constraints

Example

Response is uninformative: what happened

Q: Retrieve name from employee where age > 50

and salary < average

(retrieve salary from employee

where department = 'sales')

R: "NIL" unexpected lack of information

trace back

EMPLOYEE ← DEPARTMENT

no employee ? no department ?

CR: There is no Dept = 'sales'

Using Lexicons

A lexicon is a relation containing alternate keys,
defined by double reference pattern.

WHERE IS THE LEXINGTON?

ambiguous to a computer

not to a clerk

the WHERE can apply in the SHIPS database to
VESSELS, PORTS, or CAPTAINS

An access to the Database can resolve that question
the VESSEL='LEXINGTON' IS AT 32. . .

View Update

When the user can only perceive part of the database
then any change requested may be ambiguous.

Any update has to be fully disambiguated.

Partial disambiguation automated :

1. Formal Enumeration of alternatives
2. Reduction using Structural Constraints
3. Reduction to minimize Side-effects
4. Enumeration of candidate translators
 - a. At view definition time: by DBA selection
 - b. At execution time: using heuristic selection

Example: Assign Joe to Sgt.Peters

1. Sgt.Peters now heads Joe's unit?
2. Joe changes units

Structural knowledge:

UNIT → MEMBERS: Joe, Mike,...

UNIT → COMMAND: Sgt.Peters

∴

minimal change: 2.

Application Semantics

Two categories

- 1 Application type
- 2 Application specific

•1. Focus

Used to help completeness of response
by identifying related data

Example

Set Focus:

Transport by ship

Query:

How long does it take
to deliver 1 MT grain
to Cochin from New Orleans

Response:

Shipping time - 12 days
Average delay for port
facilities - 5 days*
Unloading time - 2 days

* This could easily be overlooked.

Specific Application Knowledge

Used to radically optimize queries [QUIST]

Example

Query:

How much oil did China export in 1982?

Knowledge:

Most oil export is by tanker

Effect:

Have query look only at tankers, rather than
at all export documents

Response:

The quantity of oil shipped by tanker from
China in 1982 is 3M barrels

Effect

Processing reduction

cargoes = oil vs. ship = tanker

25000 vs. 3000

but miss oil in barrels on freighters

Knowledge about virtual attributes

Example: Distance

We use Frames to store descriptive knowledge

A frame has a fixed number of SLOTS,
one per function expected.

a variable number of ENTRIES per Slot.

Concept Name	: DISTANCE
Value	: real
Type	: metric (for statistics)
Applies-to	: a: {VESSEL, TOWN } : b: {TOWN, VESSEL } : c: {VESSEL, VESSEL } : d: {TOWN, TOWN }
Unit	: meter
ISA	: quantity(superior frame)
Query	: How far, How distant
Computation	: if d : then sum(ROUTE-LINKS(p1,p2)): : else abs(POSITION(p1) - POSITION(p2))

Review of Knowledge Types

Structural knowledge

nearly free —

obtained when database is designed
can help in many interface problems
formality helps correctness

Application knowledge

gathered via experts —

can be added at any time
can have spectacular benefits
can introduce uncertainty

Review of Knowledge Use

The two types of knowledge
are independent of the actual content
of the database:

1. DESIGN PHASE

a formal description of the semantics
helpd define for the programmer
the Files and Access-structures

2. EXECUTION PHASE

A. At Query Parsing time - disambiguate
optimize

B. At Response Processing time
Scan Terms and Value Sets obtained for
Appropriateness

Knowledge Growth

As Queries are processed,
INFORMATION is retrieved and this can become
further KNOWLEDGE.

Before such KNOWLEDGE can enter the system
it has to be Filtered

- by Human Reasoning
 - by Deductive Validation over the Database
 - by Statistical Validation
implies Strength factors
-

Conclusion

The rapid developments of
Artificial Intelligence techniques
provide an effective approach
to deal with the awkwardness of
interfacing Large Databases

- Databases have a well-defined Scope
Within that scope Databases can be
treated as a Closed World,
considerably enhancing deductive power.

The Objective achieved with
Knowledge-Based techniques is

- move more of the well-understood
Decision-Making processes
to the computer
- reduce Programming Cost
- increase Consistency
- improve Accessibility

The KBMS Project at Stanford

Within the KBMS project we have:

- Carried out Experiments
- Demonstrated the Validity of
most of the concepts presented here.

Much work remains to be done.

- In process:

KSYS: experimental frame-based system for
Knowledge about Data
low-level frames reach into a database
frames can be members of multiple hierarchies
slots cannot be " "

Technology

Common LISP - Relational Database
Workstation

8Mb Memory, 600Mb Storage VAX750

Building a comprehensive, well-integrated system
is beyond university capabilities

We look at industry to pick up those concepts,
while we continue to attack further problems.

RELATED WORK

- Database Abstracts
- Domain Understanding
Value semantics
Distributed data management

Cooperative Responses to Boolean Queries

Francisco Corella, Symantec
S. Jerrold Kaplan, Teknowledge, Inc
Gio Wiederhold, Stanford University
Lana Yesil, Stanford Consulting Group

This work was performed at Stanford University and supported in part by NSF-IST contract 8023889 and by DARPA with the KBMS project under contract N39-80-G-0132.

ABSTRACT

Since cooperation is an essential aspect of human conversation, it is a necessary feature in a natural language interface. Previous work has shown that it is possible to incorporate principles of cooperative behaviour into a system allowing natural language access to a database. It is shown here that some of these principles can also be applied fruitfully in a formal language environment. Specifically, when a query retrieves an empty result set, it is possible, by analyzing the question and running related queries against the database, to give a useful explanation of the zero result to the user. A mathematical specification of the proper information to present to the user is described for the case where queries consist of boolean combinations of predicates of a single variable. Such cooperative responses to boolean queries have been implemented on the large bibliographic database of the Research Libraries Group. The size of the database has motivated a careful optimization of the algorithm, which is discussed.

1. Introduction

The original motivation of the research work presented here comes from previous work in computational linguistics, namely from the ideas developed for the design of CD-OP, a system for natural language access to databases ([Kaplan79]).

A major challenge in the design of a natural language (NL) interface is that, when the user is allowed to interact with the machine in English, he will expect the machine to understand and use the language as a human would. Human conversation is a powerful communication tool, which requires the full intelligence and world knowledge of human participants. Therefore the task of enabling a computer to interact, not only in natural language, but also in a "natural way", offers an unlimited field for Artificial Intelligence. However, Kaplan has shown the current feasibility of incorporating principles of language pragmatics into a NL interface to a database.

Consider the following dialog, taken from [Kaplan79], of a hypothetical NL database system with a user:

(1)

- (a) Q: Which students got a grade of F in CIS500 in Spring '77?
- (b) R: Nil. [the empty set]
- (c) Q: Did anyone fail CIS500 in Spring '77?
- (d) R: No.
- (e) Q: Was CIS500 given in Spring '77?
- (f) R: No.

If the system were more intelligent, it would be capable of the following interchange:

(2)

- (a) Q: Which students got a grade of F in CIS500 in Spring '77?
- (b) R: CIS500 was not given in Spring '77.

What is involved here is the fact that the user's question is "loaded": beyond its literal meaning, it carries the presupposition that the course CIS500 was actually given in Spring '77. The initial direct response in (1) is literally correct, but fails to rectify this wrong assumption. Upon being asked the question, a human conversant would give the indirect response (2b) instead. Failing to do so would violate the tacit conventions of cooperation in human conversation. Not only would the answer be insufficiently informative, it would actually reinforce the incorrect presupposition of the questioner.

Thus (1b) is not only poor, it is misleading.

We shall categorize answers such as (2b) as CORRECTIVE INDIRECT RESPONSES. Sample dialogs between CO-OP and human users, showing such responses, are given in [Kaplan79].

A second type of indirect responses that were explored in CO-OP, and are relevant to the work presented in this paper, are SUGGESTIVE INDIRECT RESPONSES. We borrow again an example from [Kaplan79] in order to explain this notion:

(3)

(a) Q: Are there any seats available in the orchestra for tonight's Rolling Stones concert?

(b) R: No.

(c) Q: Are there any in the balcony?

(d) R: Yes.

(4)

(a) Q: Are there any seats available in the orchestra for tonight's Rolling Stones concert?

(b) R: No, but there are some in the balcony.

The answer "No" in (3) is literally correct, but a ticket office clerk will, in general, give (4b) instead. This answers not only the initial query, but also the follow-up query:

"Are there any in the balcony?"

Of course, it is not a simple matter to decide what the anticipated follow-up query, and the corresponding suggestive indirect response should be. Here, the second question derives from the first one by shifting the focus of the query from "orchestra seats" to "balcony seats". Thus, giving such an indirect response requires determining the focus of the query. In CO-OP, the focus was selected using a set of heuristic rules, and the suggestive response was obtained by enlarging the focus (rather than shifting it).

The possibility of asking questions to a database in NL is very attractive, and several implementations have shown the feasibility of the approach. However, most databases today do not have a NL interface. On the other hand, even when NL systems are available, a formal query language will still provide a more appropriate access method for the experienced user.

Therefore, having investigated the principles of cooperative behaviour

in a NL setting, the following questions arise naturally:

- (1) Do the principles of cooperative discourse carry over to a formal query language environment?
- (2) Are indirect responses useful in a formal language setting?
- (3) When should indirect responses be given?
- (4) What information should they present to the user?

In order to explore these questions, we have selected a formal query language and a specific database, the bibliographic database of the Research Libraries Group (RLG).

2. The RLG database

2.1. Bibliographic files and indexes

The RLG computer system serves as a common on-line "card catalog" for a group of libraries, mainly belonging to academic institutions. The data are centralized as a Spires database system ([SPIRES73]) running under DRVYL on an IBM mainframe, located at the Center for Information Technology, Stanford University. More than 200 simultaneous users are currently supported.

The bibliographic records are grouped in 6 files, as follows:

(data compiled in November 1981)		
Books	4,479,000	records
Serials	851,000	"
Recordings	41,000	"
Scores	15,000	"
Films	58,000	"
Maps	70,000	"

Some facilities provided by the system make concurrent use of several of these files. For instance, the system will run the same query on a predetermined sequence of files if the user so desires. All the examples in this paper will be taken from the Books file.

Individual records are organized according to standard bibliographic criteria as a set of precisely defined fields. The reader who is not familiar with the complexities of cataloguing work will probably be surprised to know that there are on the order of one thousand such fields. Of course only a small subset of them is used in each record, which is internally organized as a hierarchical structure of optional and variable length fields.

Fortunately, this complexity is nicely packaged by a set of indexes (18 shared indexes and 10 local indexes (IRLGB1)). Each index can be viewed as a two-place predicate, whose first argument is a bibliographic entity, represented by a record in the database, and whose second argument is a string. There is, for instance a Title Word (TW) index. The predicate,

TW(X, "sunset")

where X is variable ranging over one of the files, will be true if the string "sunset" appears as a separate word in one of a well defined set of bibliographic fields, including:

- title
- sub-title
- parallel title
- title statement
- uniform title
- foreign language translation of title

and many others, up to a total of 259 different fields.

Internally, an index is organized, at least conceptually, as a list of <key, pointer-list> pairs. One of the pairs in the TW index will have as a key, the word "sunset" (for example), and an associated list of pointers to all the records X for which the predicate

TW(X, "sunset")

has the value TRUE, i.e. all the records that have the word SUNSET in one of the 259 title-related bibliographic fields.

Another useful index is the Personal Name index (PN). It acts on a variety of fields related to authorship of bibliographic materials, and incorporates a personal name algorithm that takes into account the multiple ways in which a given name can be written (IRLGB1). For example, both predicates

PN(X, "M. de Cervantes Saavedra")
PN(X, "Cervantes, Miguel")

will be true for a record X representing the book "Don Quijote de la Mancha" whose author is:

Miguel de Cervantes Saavedra.

2.2. Boolean queries

The query language used is propositional calculus. The user specifies the result set he wishes to retrieve as the set of all records X that satisfy a given predicate, and he can express this predicate as an arbitrary boolean combination of atomic index predicates, using the operators:

AND,
OR,
AND NOT.

The keyword FIND is used as the first word of the command, and, since only one variable is involved, X is not mentioned.

Thus, if we are looking for the book:

"Sunset back roads of California",
by Earl Thollander

we may specify the result set:

{X \ TW(X, "sunset") AND TW(X, "roads") AND PN(X, "Thollander")}

with the query:

FIND (TW="sunset") AND (TW="roads") AND (PN="Thollander")

Some typing is saved by omitting quotes, equal signs, parenthesis, the conjunction operator (AND is assumed when no operator is specified) and even some of the index names (which are sticky). Thus we may simply type:

find tw sunset roads pn thollander

If we are not sure whether the title mentions ROADS or TRAILS, we may ask:

find tw sunset (trails or roads) pn thollander

And if we want any other book by the same author:

find pn thollander and not (tw sunset (roads or trails))

Parenthesization allows in theory to specify an arbitrarily nested boolean expression as the set-former condition. The query is represented internally by its expression tree. The leaves of the tree are index predicates (such as TW="sunset"). These atomic predicates will be called from now on PRIMARY SEARCH TERMS. To each of them corresponds a PRIMARY RESULT SET, materialized by the pointer-list stored in the corresponding index node. The result set of the query could be obtained by computing the intermediate result sets for each node of the expression tree. If the node operator is AND, then its associated result set is the intersection of the result sets of its children, if OR, the union, if AND NOT, the set difference. However, because result sets can be quite large, intermediate results are not

actually computed. Instead, a parallel search is performed on the index nodes corresponding to the leaves of the tree.

The reader may have noticed the special treatment of negation. There is no NOT operator, and this is compensated by the presence of a non-commutative AND NOT. It is worthwhile analyzing the motivation behind this choice of logical connectives. Let us consider a query that would not be subject to this restriction, and let D be the union of its primary result sets, R the set of all the records in the file, and D' the complement of D with respect to R . The boolean condition can be expressed in disjunctive normal form, i.e. as a disjunction of conjunctions (OR of AND's) of primary search terms, some of them negated. The result set of a primary search term is included in D , therefore the result set of a negated primary search term contains D' . If at least one of the terms of a conjunction is not negated, the result set of the conjunction is included in D . If all the primary search terms of a conjunction are negated, then its result set contains D' . Now, if at least one of the conjunctions has all its terms preceded by NOT, the result set of the query, being the union of the result sets of the conjunctions, contains D' . Otherwise (if every conjunction has at least one non-negated primary search term) the overall result set is included in D .

This suggests that two types of queries be distinguished:

- (1) Queries whose result set is included in D .
- (2) Queries whose result set contains D' .

R having on the order of 5 million records for the Books file, a result set of a query of type (2) can be considered to be "infinite" for all practical purposes. In any case, a query of type (2) will not yield a useful result set. Now, a query of type (1) can always be expressed with AND NOT instead of NOT operators, simply by placing one of the non-negated primary search term as the first term of each conjunction. Reciprocally, it is easy to show, by induction on the number of nodes in the expression tree, that, when a boolean expression with connectives AND, OR, AND NOT is put in disjunctive normal form, every conjunction involves at least one non-negated primary search term. Therefore, the queries that can be expressed with the provided connectors are exactly those of type (1).

3 Cooperative responses to boolean queries

3.1. Search request patterns

Librarians in the network libraries use the RLG database when they order, receive and catalog bibliographic materials, and when they give reference services to the public. The variety of search requests that are issued during the execution of those tasks can be classified in two broad categories:

(1) Requests that aim at a single item and specify a code that uniquely identifies it. This code may be the Library of Congress Card Number, International Standard Book Number, library call number, or other.

(2) Requests that aim at a single or a small group of items, and are based on partial information, usually consisting of title words and/or author names.

Requests of type (2) are susceptible of cooperative responses and have been the focus of our work. They are essential to all phases of the librarian's work, and will become even more important in the very near future when library patrons are given access to the database through public terminals.

A request of type (2) is successful when it retrieves a result set consisting of one or a small group of items. A result set which is either empty or too large (e.g. more than one hundred records), is often due to an inadequate specification. In general, the user will then issue a series of follow-up queries, until either he is able to closely circumscribe the set he wants, or he ascertains that the desired items are not present in the database.

The system keeps the result set of the last query that has been issued. If the user starts a query with an operator, instead of the word FIND, the left operand will be the result of the previous query. Thus a large result set can be restricted, and may be a step in the refinement of a search request. An empty result set (a "zero result" in the local professional jargon), however, amounts to a failure.

3.2. Zero results

It seems therefore that we have already obtained the answer to the third question asked in the introduction: if cooperative responses are useful at all in the RLG system, then it is after a zero result that they should be helpful. It was also the case in CO-OP that null results were the trigger of cooperative responses. What else carries over from a NL environment to the RLG formal language system?

Consider the query:

Find (tw sunset trails) and (pn Thollander)

It retrieves 4 entries ("clusters") corresponding to the book:

Sunset back roads of California, by Earl Thollander

as shown in figure 1.

Now imagine that the user does not remember the title of the book well

BKS/PROD Books MUL Search CRLG-INT
FIN (TW SUNSET ROADS) AND (PN THOLLANDER) - 4 Cluster(s) in PBKS

1) Thollander, Earl. BACK ROADS OF CALIFORNIA : 2d ed. (Menlo Park, Calif. :
Lane Pub. Co., c1977.)
DCLC (c-9110 DLC) CRPG (c-9610 CRic) CSFX (c-9610 CSf) CUBG (c-9610 CU)
CUDG (c-9610 CU-A) ILEG (c-9610 IE) AZFG (c-9610 AzF) .

2) SUNSET BACK ROADS OF CALIFORNIA. (2d ed. Lane 1977)
CSCG (c-9660 CSJCL)

3) Thollander, Earl. SUNSET BACK ROADS OF CALIFORNIA : 2d ed. (Menlo Park,
Calif. : Lane Pub. Co., c1977.)
CH2G (c-9610 CStrN3)

4) Thollander, Earl. BACK ROADS OF CALIFORNIA; (Menlo Park, Calif., Lane
Magazine & Book Co. [1971])
NYPG (c-1665 NN) AZFG (c-9610 AzF) CSUG (c-9610 CSt) CSUU (c-9667 CSt)
DCLC (c-9110 DLC)

- Figure 1 -

Each library has a record for each edition of the book that it owns,
and records for the same edition of the same book owned by different
libraries are "clustered" in order to keep down the size of the
indexes and simplify the displays. The 4-letter codes appearing after
each entry (DCLC, CRPG, CSFX, ...) are the ID's of the libraries that
own records in the cluster.

enough and issues the query:

Find (tw sunset trails) and (pn Thollander)

This query retrieves an empty set. But there is a fundamental difference with the NL examples given in section 1: the query does not contain any presuppositions. All three predicates:

- (TW="sunset")
- (TW="trails")
- (PN="Thollander")

have the same role in the query. When using a formal language, we may gain in expressive power within the limited domain for which the language has been designed, but we also may lose the richness of nuance of human language.

However, if the query were addressed to a hypothetical human being that had complete knowledge of the contents of the database, the answer would not be simply "no books found". The database-being, if friendly, could say that there are a number of books (namely 593 clusters) with the word "trails" in their titles. Or he could say that there are 24 clusters with author name "Thollander", and that 4 of them have the word "sunset" in their titles, if not the word "trails".

Recognizing the relationship between "roads" and "trails" is completely out of the scope of the work presented here. However, it happens that the four clusters of interest can be retrieved by the query:

Find (tw sunset) and (pn Thollander)

This query is the conjunction of two predicates,

- (TW="sunset")
- (PN="Thollander")

which appear in the query of the user. We shall say that a query Q2 which consists of a boolean combination of primary search terms, which all appear in a query Q1, is a subquery of Q1. Thus both:

Find (tw trails)

and

Find (tw sunset) and (pn Thollander)

are subqueries of

Find (tw sunset trails) and (pn Thollander)

In general, we can say that the result sets of the subqueries contain information which is potentially of interest to the user. We cannot display these result sets, because the user would be flooded with information. But we have found that the cardinalities of some of these sets can give very good clues to help determine the cause of the zero result, and can suggest the right follow-up queries.

Even after having decided to present the number of items in the sets rather than the sets themselves, this still represents more information than can be usefully displayed. Also, much of this information is irrelevant or redundant. We therefore have to select the appropriate subqueries whose result set cardinalities should be presented. In the next 2 sections we show that it is possible to determine exactly which subqueries these are. For a given zero result query, they constitute a small set, and the corresponding cooperative response is such that the user can pick up at a glance the critical information.

3.3. Conjunctive queries

Let us first center the discussion on purely conjunctive queries, those that make use only of the operator AND. Let us represent by S, T, A the three predicates of the conjunctive query that we have been using as an example.

Find (tw sunset trails) and (pn Thollander)

```
S      ==      (TW-"SUNSET")
T      ==      (TW-"TRAILS")
A      ==      (PN-"THOLLANDER")
```

As described in section 3.2, a cooperative response to this zero result query should give the cardinalities of the results of a set of subqueries, which have to be selected. For now, we shall limit ourselves to conjunctive subqueries (section 3.4 will present a general discussion).

Each conjunctive subquery corresponds to a subset of

$$E = \{S, T, A\}$$

The power set of E, $P(E)$, has been represented as a graph in figure 2. The node ST, for instance, corresponds to the subquery.

Find (tw sunset) and (pn Thollander)

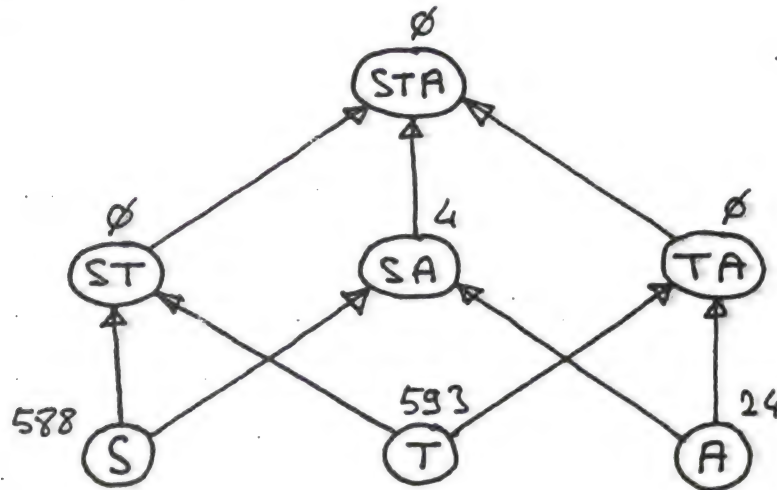
The arcs of the graph represent the set-inclusion relationship in the power set. Let B be the Books file, considered to be a set of database entities, (clusters in this case) and $P(B)$ its power set. We define R to be the function from $P(E)$ onto $P(B)$ that maps a subset of $\{S, T, A\}$ onto the result

Find (tw sunset trails) and (pn Thollander)

S == (TW="SUNSET")

T == (TW="TRAILS")

A == (PN="THOLLANDER")



- Figure 2 -
Conjunctive Subqueries

BKS/PROD Books

Search

CRLG-INT

FIN (TW SUNSET TRAILS) AND (PN THOLLANDER) - None in PBKS.

In file PBKS there are:

- 593 clusters with (TW="TRAILS")
- 4 clusters with (PN="THOLLANDER") AND (TW="SUNSET")

but none with:

- (TW="TRAILS") AND (TW="SUNSET")
- (PN="THOLLANDER") AND (TW="TRAILS")

- Figure 3 -
Cooperative response to a purely conjunctive query

set of the corresponding subquery. R is an homomorphism from the partial ordering established by set-inclusion in the power-set $P(E)$ onto the inverse of set-inclusion in $P(B)$, i.e. if E' and E'' are subsets of E , then

$$E' \text{ included in } E'' \Rightarrow R(E'') \text{ included in } R(E')$$

(the reciprocal is not true). The empty set has not been represented in figure 2. Its associated result set, $R(\emptyset)$, would be B itself.

An element U of a subset W of $P(E)$ is a minimal element of W if there are no elements of W strictly included in U . In the same way, U will be said to be a maximal element of W if no elements of W strictly contain U .

The nodes of figure 2 are labeled by the cardinalities of the associated result sets. Our problem is to determine which of these labels should be presented to the user. The label 0 on STA means that the initial query has a zero result. This is the direct response to the query. But we can see that there are 2 other zero results in the diagram, on the nodes ST and TA. Because R is an homomorphism, each one of the facts

$$\begin{aligned} \text{card}(R(ST)) &= 0 \\ \text{card}(R(TA)) &= 0 \end{aligned}$$

implies

$$\text{card}(R(STA)) = 0.$$

In other words, saying that there are no clusters with

$(TW="SUNSET") \text{ AND } (TW="TRAILS")$

is more precise, carries more information, than saying that there are no clusters with

$(TW="SUNSET") \text{ AND } (TW="TRAILS") \text{ AND } (PN="THOLLANDER")$

We have therefore already found an indirect answer that would be better than the direct one, namely:

" In the books file there are no clusters with:
 - $(TW="SUNSET") \text{ AND } (TW="TRAILS")$
 - $(TW="TRAILS") \text{ AND } (PN="THOLLANDER")$ "

This will give the user a clue to the explanation of the zero result. TRAILS conflicts both with SUNSET and THOLLANDER, while SUNSET and THOLLANDER do not conflict with each other.

In general, if U and V are two subqueries, giving zero results, and if U is included in V (as a subset of E), then the fact that V gives a zero result is an obvious consequence of the fact that U gives a zero result. Therefore

the only zero results to be presented to the user are those that are MINIMAL ELEMENTS of the set of subqueries giving zero results.

Let us turn now to the subqueries that give non-zero results. The fact that there are 4 clusters with

(TW="SUNSET") AND (PN="THOLLANDER")

is certainly a most valuable piece of information to be given to the user. Figure 2 also shows that there are, in the Books file, 24 clusters with (PN="THOLLANDER"). Should this fact be included in the response? In his query, the user has expressed interest on books by Thollander. The result of the query being empty, it could be that there are no books by Thollander at all included in the database (for example, the name could have been misspelled). Therefore the fact that there are actually entries with (PN="THOLLANDER") is certainly of interest to the user. But this we know already when we know that the combination of search terms (TW="SUNSET") and (PN="THOLLANDER") yields a non-zero result.

Now, for non-zero nodes, we shall be giving the cardinalities of the result sets. Is it useful to present the fact that there are exactly 24 clusters with (PN="THOLLANDER"), in addition to the fact that there are 4 with (TW="SUNSET") AND (PN="THOLLANDER")? By introducing the restriction (TW="SUNSET"), the user has explicitly requested the removal from the result set of books that do not have the word sunset in their title. Since there actually are books by Thollander with "sunset" in their title, there is no point in letting the user know how many other books by Thollander there are.

In general, if U and V are two subqueries giving non-zero results, and if U is included in V (as a subset of E), then the cardinality of U should not be included in an indirect cooperative response. Therefore the only non-zero results whose cardinalities are to be presented to the user are those that are MAXIMAL ELEMENTS of the set of subqueries giving non-zero results.

We can summarize the two rules that we have found in a clear-cut specification of the information to be given to the user when a purely conjunctive query gives a zero result:

PRESENT THE MINIMAL ZERO RESULTS AND THE CARDINALITIES OF THE MAXIMAL NON-ZERO RESULTS.

This leads in our current example to the following indirect response:

" In the books file there are:
- 593 clusters with (TW="TRAILS")
- 4 clusters with (TW="SUNSET") AND (PN="THOLLANDER")

but none with:
- (TW="SUNSET") AND (TW="TRAILS")
- (TW="TRAILS") AND (PN="THOLLANDER") "

Figure 3 shows the actual appearance of the screen.

Notice how, while the negative part of the answer gives clues to an explanation of the zero result, the positive part of it suggests possible follow-up queries. Furthermore, the cardinalities of the maximal non-zero results help discriminate between them. Thus, here, the low cardinality of the second one designates the query:

"Find (tw sunset) and (pn Thollander)"

as a good candidate to be the next query.

3.4. General boolean queries

Let us consider now two examples of queries involving the operators OR and AND NOT:

(Q1) "Find pn Thollender tw sunset (roads or trails)"

(Q2) "Find pn Thollander tw sunset and not (tw back roads)"

Both give zero results, the first one because of the spelling mistake in the author name, the second one because "Sunset back roads of California" is the only book by Thollander with the word "sunset" in its title.

There are many equivalent ways to express a query, the laws of boolean algebra providing a formalization of the rules to go from one form to another. If we want to devise a cooperative response that addresses the meaning of the query, and is independent of the particular form being used, it is a good idea to start by reducing the query to some equivalent canonical form. Disjunctive and conjunctive normal form are candidates for that role.

When a disjunctive query yields a zero result we know that each one of the terms of the disjunction will also yield a zero result. On the other hand, when a conjunctive query has an empty result, we can say nothing about the terms of the conjunction. This strongly suggests using disjunctive normal form (DNF) as the tool for the analysis of a zero result query. Let us therefore transform our two example queries into DNF.

We shall use additive notation for disjunction, multiplicative notation for conjunction, \sim to denote the negation operator, and the following abbreviations for the primary search terms:

A	==	(PN-"THOLLANDER")
E	==	(PN-"THOLLENDER")
S	==	(TIJ-"SUNSET")
B	==	(TW-"BACK")

R = (TW "ROADS")
 T == (TW-"TRAILS")

The boolean expressions of Q1 and Q2 can be transformed as follows:

Q1:
 $E S (R + T) = E S R + E S T$

Q2:
 $A S \sim(B R) = A S (\sim B + \sim R)$
 $= A S \sim B + A S \sim R$

We have now to determine what subquery cardinalities should be incorporated in the cooperative response to a general boolean query.

Recall that by "subquery" we mean any query made up of primary search terms that appear in the user's query. The number of classes of equivalent subqueries is large, but most of them are of no interest. We are going to get rid of those, and restrict the space of relevant subqueries in four successive steps.

(1) When we transform an expression into DNF, we can proceed in two steps. First we push negations down to the leaves of the expression tree by repeated application of De Morgan's laws. Then, using the property of distributivity of AND with respect to OR, we break conjunctions into disjunctions repeatedly until we reach DNF. We shall say that an expression in which negations have been pushed to the leaves of the tree is in De-Morgan form. In a De-Morgan expression, a leaf of the tree (primary search term, PST) is NEGATED if its parent is the operator NOT, ASSERTED otherwise. The example query Q1 is already in De-Morgan form, all its PST's being asserted. Query Q2 is not in De-Morgan form. The two steps of the transformation into DNF are those given above. After the first step we obtain the De-Morgan expression:

$A S (\sim B + \sim R)$

where A and S are asserted while B and R are negated.

It is clear that, when a De-Morgan expression is transformed into DNF by AND/OR distributivity, the association of NOT operators with leaves of the tree does not change. Therefore, if a PST is negated in a De-Morgan expression it will also be negated in its DNF, and hence, in any other equivalent De-Morgan expression. If a PST is asserted in a De-Morgan expression, it will also be asserted in any other equivalent De-Morgan expression. This allows us to define the signature (+/-) of a PST A in an expression Q as the fact that A is asserted/negated in any De-Morgan expression equivalent to Q. We shall say that a subquery U of Q is compatible with Q if each PST of U has the same signature in U as in Q.

As the first step in the pruning of the subquery space, we shall decide to consider only compatible subqueries. We shall let the reader convince himself of the validity of this step.

(2) Figures 4 and 5 represent the space of all possible combinations of signed PST's for the queries Q1 and Q2. Each node therefore represents a CONJUNCTIVE compatible subquery and is labeled with the cardinality of the corresponding result set. Each compatible subquery (not necessarily conjunctive) can be represented by a thin set of such nodes, the nodes of the set corresponding to the terms of the DNF. (By thin set we mean any set which does not include a pair of nodes linked by an arc).

The second pruning step consists of restricting ourselves to considering only conjunctive compatible subqueries. The justification for this step is that the cardinalities of all the compatible subqueries can be inferred from those of the conjunctive ones. Indeed, if Q is a compatible subquery and T1 and T2 are the terms of its DNF, then

$$\text{card}(R(Q)) = \text{card}(R(T1)) + \text{card}(R(T2)) - \text{card}(R(T1 * T2))$$

where $T1 * T2$ is also a conjunctive compatible subquery, namely the conjunction of the signed PST's appearing either in T1 or T2. If the DNF of Q has three terms T1, T2, T3, then

$$\begin{aligned} \text{card}(R(Q)) = & \text{card}(R(T1)) + \text{card}(R(T2)) + \text{card}(R(T3)) \\ & - \text{card}(T1 * T2) - \text{card}(R(T1 * T3)) - \text{card}(R(T2 * T3)) \\ & + \text{card}(T1 * T2 * T3) \end{aligned}$$

Similar formulas hold when the DNF of Q has 4 or more terms. Therefore we do not lose any cardinality information by considering only conjunctive compatible subqueries, i.e. those that correspond to single nodes in figures 4 and 5.

Notice, in particular, that the result set of a query is empty if and only if each one of the result sets of the subqueries corresponding to the terms of the DNF is empty.

(3) In figures 4 and 5, the terms of the DNF of the query have been doubly circled. In each case we shall define the "area of interest" of the query as the set of nodes which are included in at least one of these terms. The DNF terms are the maximal elements of the area of interest.

When a query gives a zero result, each one of the DNF terms gives itself a zero result. The nodes in the area of interest correspond to the conjunctive compatible subqueries that we obtain from the terms of the DNF by relaxing constraints. Common factors can be reached from more than one term, and are represented in the graph by nodes, such as ES in figure 4, which are included in more than one maximal element of the area. In figures 4 and 5,

Find pn Thollender tw sunset (roads or trails)

E == (PN="THOLLENDER")
 S == (TW="SUNSET")
 R == (TW="ROADS")
 T == (TW="TRAILS")

$$E S (R + T) = E S R + E S T$$

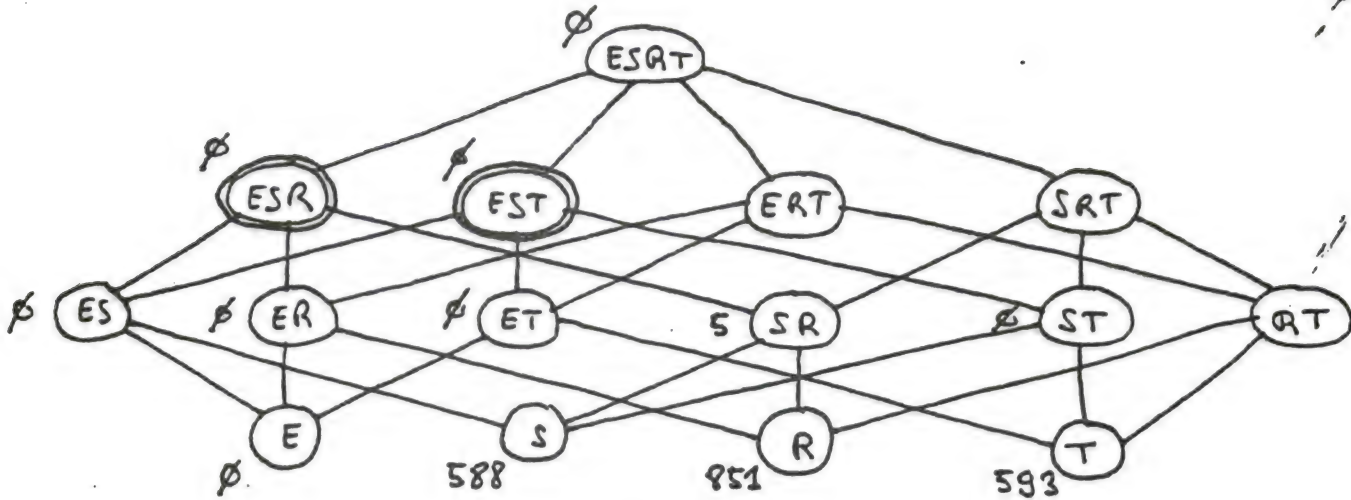


Figure 4
 Conjunctive compatible subqueries for Q1

Find pn Thollender tw sunset and not (tw back roads)

A == (PN="THOLLANDER")
 S == (TW="SUNSET")
 B == (TW="BACK")
 R == (TW="ROADS")

$$A S \sim (B R) = A S (\sim B + \sim R) \\ = A S \sim B + A S \sim R$$

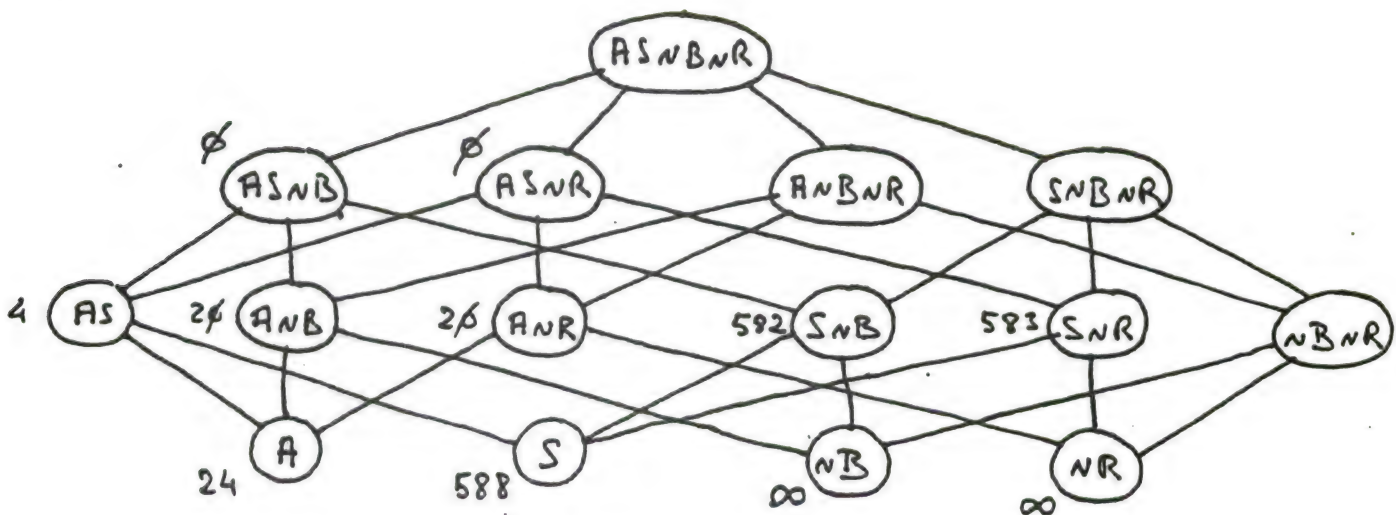


Figure 5
 Conjunctive compatible subqueries for Q2

nodes in each area of interest have been labeled by the cardinalities of their result sets.

The third step in the pruning process will restrict the set of subqueries to those in the area of interest. This excludes subqueries like RT in figure 4, which refers to books having both words "roads" and "trails" in their titles.

(4) Finally, inside the area of interest, the same considerations apply as in section 3.3. We distinguish between zero and non-zero results. The direct zero-result response to the query is equivalent to giving the terms of the DNF as zero results. Giving instead the minimal non-zero results in the area of interest is a more precise answer, which explains the cause of the failure. Giving the maximal non-zero results in the area of interest, together with the cardinalities of the associated result sets, will suggest useful follow-up queries. As an exception, when a maximal non-zero result consists only of negated primary search terms, it is not necessary to mention it in the answer. The user knows that its cardinality is of the order of magnitude of the file itself. Considering these sets to be "infinite" for practical purposes we can summarize the rules that determine the contents of the response as follows:

GIVE THE MINIMAL ZERO RESULTS AND THE CARDINALITIES OF THE MAXIMAL "FINITE" NON-ZERO RESULTS.

The answers to queries Q1 and Q2 are given in figures 6 and 7.

4. Implementation notes for a large database.

As we have seen in section 2, the books file in the RLG database has 4.5 million records. Primary search terms often yield result sets with several hundred or several thousand records. In addition, the database is being queried simultaneously by more than 200 users. In such an environment, efficiency considerations are critical.

In order to minimize the load on the system, the algorithm must satisfy the following implementation principles:

(1) The result sets should not be recomputed from scratch for each subquery. If subqueries are enumerated in an order such that those with a smaller number of primary search terms are visited first, then it is possible to compute the result set of a conjunctive subquery having $(n + 1)$ PST's as the intersection of two result sets of n -PST subqueries, which have been already computed. More precisely, we can choose any two nodes with n PST's which are included in the $(n + 1)$ -PST node of the initial subquery. As a refinement we can select those whose result sets have the smallest cardinality.

(2) In order to satisfy the above principle, we must keep intermediate result sets after they have been computed. Having large result sets in core

BKS/PROD Books Search CRLG-INT
FIN (PN THOLLANDER) AND (TW SUNSET) AND (TW ROADS OR TRAILS) - None in PBKS

In file PBKS there are:

- 593 clusters with (TW="TRAILS")
- 5 clusters with (TW="ROADS") AND (TW="SUNSET")

but none with:

- (TW="TRAILS") AND (TW "SUNSET")
- (PN="THOLLANDER")

- Figure 6 -
Cooperative response to query Q1

BKS/PROD Books Search CRLG-INT
FIN (PN THOLLANDER) AND (TW SUNSET) AND NOT (TW BACK ROADS) - None in PBKS

In file PBKS there are:

- 4 clusters with (TW="SUNSET") AND (PN="THOLLANDER")
- 20 clusters with (PN="THOLLANDER") AND NOT (TW="BACK")
- 20 clusters with (PN="THOLLANDER") AND NOT (TW="ROADS")
- 582 clusters with (TW="SUNSET") AND NOT (TW="BACK")
- 583 clusters with (TW="SUNSET") AND NOT (TW="ROADS")

but none with:

- (TW="SUNSET") AND (PN="THOLLANDER") AND NOT (TW="BACK")
- (TW="SUNSET") AND (PN="THOLLANDER") AND NOT (TW="ROADS")

- Figure 7 -
Cooperative response to query Q2

increases swapping of virtual memory pages. Therefore it is important to determine the exact moment at which an intermediate result set will not be needed any more and can be released.

(3) If a node A is included in a node B, and if A yields a zero result, then we know that B will yield a zero result. It is important to avoid, not only the computation of $R(B)$, but even the enumeration of subquery B. In the same way, if A is not in the area of interest, then we know that B is not in the area of interest either, and we avoid the computation of $R(B)$ and even the enumeration of B. Thus large areas of the subquery space are not explored at all, and the pruning steps described in 3.4 translate into an efficient implementation, in addition of providing more concise responses.

The core of the implementation is the algorithm used for enumerating the subqueries. The technique that has been used is based in the following theorem.

Definitions:

In a partial ordering, X is said to be an upper bound of a subset S if it is greater than every element of S; Y is said to be the least upper bound of S if it is an upper bound and is less than any other upper bound. Let us define an upperbound lattice to be a partially ordered set where any pair of elements has a least upper bound. Let us also define the lattice generated by a subset to be the smallest upperbound lattice that contains the subset.

Theorem:

If E_1, E_2, \dots, E_p are elements of S, the lattice generated by $\{E_1, E_2, \dots, E_p\}$ is the union of:

- the lattice generated by $\{E_1 * E_2, E_1 * E_3, \dots, E_1 * E_p\}$, where $A * B$ designates the least upper bound of the pair $\{A, B\}$
- the singleton $\{E_1\}$
- the lattice generated by $\{E_2, E_3, \dots, E_p\}$

Since both $\{E_1 * E_2, E_1 * E_3, \dots, E_1 * E_p\}$ and $\{E_2, E_3, \dots, E_p\}$ have $(p - 1)$ elements, the theorem leads to a straightforward recursive algorithm for enumerating the elements of the lattice generated by a finite set. Moreover, a second part of the theorem asserts that, in the case where none of the elements of the set is a member of the lattice generated by the other elements (i.e. the elements are "independent"), each element of the lattice is generated only once.

The set of the conjunctive compatible subqueries of a given query is

partially ordered by set-inclusion, if we consider a subquery to be represented by the set of its signed primary search terms. A pair of subqueries has a least upper bound which is the union of the subqueries, again considered to be sets of PST's. The recursive algorithm suggested above does not yield the subqueries in the order desired. However, a more careful nonrecursive implementation based on the same principle (not described here), does enumerate them in ascending order with respect to the number of PST's, and meets the implementation requirements listed above.

5. Directions for further research.

It seems now that in the context of the RLG database, we have precise answers to all four questions asked in section 1:

(1) The principles of cooperative discourse, and in particular the need for the general detection and correction of erroneous presuppositions in a query, do not entirely carry over to the RLG formal language environment.

(2) However, indirect responses are useful in explaining zero results and suggesting follow-up queries.

(3) Indirect responses should be given when a query yields an empty result set.

(4) The information to be presented to the user has been specified in section 3.

Points (2) and (3) seem to have general validity. But the answers to (1) and (4) are specific to the particular context of the RLG database. With a richer data model, and a more complex query language, allowing quantification, it is conceivable that queries may carry presuppositions on the state or the structure of the database, and, of course, the specification given here of the information to be presented in a cooperative response will not carry over. An attempt to specify precisely this information in the context of a richer, but still self-contained and well-defined formal-language environment, would be the logical extension of the work presented in this paper.

6. Acknowledgments

The authors greatly appreciate the opportunity they were given to carry out this research project on the bibliographic database of the Research Libraries Group, Inc., and the cooperation and support received from John Schroeder, David Richards, Glee Cady and other members of the R.L.G. staff.

7. References

[Kaplan79] Kaplan, S. J.
Cooperative Responses from a Portable Natural Language Data Base Query
System, Ph.D. dissertation, Dept. of Computer and Information Science,
University of Pennsylvania, 1979.

[RLGB1] The Research Libraries Group, Inc.
Searching in RLIN II. User's Manual. 1981.

[SPIRES73] SPIRES project
Design of the Stanford Public Information Retrieval System (Spires II)
Stanford Center for Information Processing, Stanford University, 1973.

[Wiederhold81] Gio Wiederhold, S. Jerrold Kaplan, Daniel Sagalovicz
Research in Knowledge Base Management Systems
ACM Sigmod Record Vol. 11 no. 5, April 1981, p.26-54

AUTHOR AFFILIATION: Gib Wiederhold is an Associate Professor of Medicine and Computer Science (Research) at Stanford University.

ADDRESS: Computer Science Department
Stanford University
Stanford, CA 94305

WHAT DOES IT MEAN FOR AN INFORMATION SYSTEM INTERFACE TO BE 'INTELLIGENT'?

Nicholas J. Belkin

An information system in general consists of a user, a knowledge resource, and an intermediary mechanism between user and knowledge resource. In most current information retrieval systems, the intermediary includes a rather large human component. Attempts to provide direct end-user access to the knowledge resource; that is, to automate wholly the intermediary component, have taken a rather narrow view of the role of the intermediary in the information system, and in general a highly idealized view of the nature of the user's situation. If computer intermediary or interface is to be considered intelligent (and indeed for it to be more than superficially useful), it must perform the functions that human intermediaries find necessary to successful interaction. This implies that we require analysis of the role and functions of the intermediary in the information system, especially in terms of its interaction with the user. This paper presents such an analysis, and suggests a research framework and design specification for intelligent information system interfaces.

Information Retrieval, Intelligent Interfaces

1. The information system situation

The situation with which we are concerned is direct end-user searching in computerized information systems in general. This information system situation can be characterized as follows:

a user, with goals and intentions, recognizes that her/his state of knowledge is inadequate (anomalous) with respect to the goals, or to some problem;

to resolve the anomaly, the user has recourse to some information provision mechanism (IPM), which consists of a knowledge resource, and an intermediary access mechanism.

The information system, then, is a recipient-controlled communication system, consisting of interaction among the following participants:

the user (recipient), with a problem, goals, intentions, knowledge, beliefs;

the knowledge resource, which is, in general, a set of texts, represented and organized in some way;

the intermediary, which interacts with the user and the knowledge resource, whose role is mediating between the user and the knowledge resource, and which brings to the interaction her/his own goals, intentions, beliefs and knowledge.

Typical information systems which fit within this definition include student advisory services, social security benefit offices, decision support systems and document retrieval systems. Because our research has concentrated upon the last type of information system, we will use it here as our example, but we wish to stress that we have also investigated other information system types, and that there is some generality among them all.

2. Users in information systems

There are a number of significant characteristics of users in the sorts of information systems which we have described. In particular, they are usually intermittent or irregular participants in such systems, such participation not being their sole, or even usual activity. Related to this characteristic, such users are often difficult to identify or characterize, and although there may be one or a few dimensions along which they are similar, in general there are many more on which they are heterogeneous. Furthermore, the problems which they bring to the system are usually ill-defined, and their knowledge about the situation is, typically, at least perceived as inadequate.

The last characteristic is particularly important, for it implies that they lack that very knowledge which is required for them to specify what information is required to modify their states of knowledge appropriately. (Of course, some information problems are such that requirements can be specified, but our research, and that

of others, has shown that this is the case much less often than had been supposed) This means, in general, that asking the user to specify how the state of knowledge ought to be modified; that is, asking her/him to specify an information requirement, is not an appropriate strategy. Yet if we look at how operational (and indeed many experimental) information systems appear to be constructed, this strategy seems to be the basis of their design. But these systems (at least in the document retrieval environment) all require human intermediaries. It certainly appears to be worthwhile trying to discover and understand just what it is that these people do, in order to find out how it is that the systems manage to achieve their goals to even the levels that are accomplished.

3. What happens in systems with human intermediaries?

If information systems appear to work well only when there is a significant human intermediary component, then it seems reasonable to ask what the human intermediary does, especially if we wish to design systems which have automated, rather than human intermediaries. Figures 1 and 2 are excerpts from transcripts of interactions between human intermediaries (I) and end users (U) in an operational online document retrieval service. These transcripts demonstrate, as has earlier research by Taylor (1968), that one thing that usually happens is that intermediaries and users engage in substantial interaction aimed at the intermediary's obtaining a model of the user's goals, problem and situation in general (what Wersig (1979) has called the 'problematic situation'). That is, rather than asking the user to specify a search topic (information requirement), the intermediaries first try to gain an understanding of the context in which any eventual search topic will be embedded.

I Alright. right the form ... err, what we got on the form
U

I just says community education in developing countries (.) that's
U yeah

I approximately yeah/1 can you tell me (,) sort of quite
U yeah well (illeg)/2

I a lot more about what it is you're going to do - it is your
U

I dissertation/3 yeah yeah (.) tell me what sorts
U yeah M.A. dissertation/4

I of things you're going to do in your dissertation (,) and then the
U

I sorts of things you want to rea::d/5
U

Figure 1. Partial transcript of user-intermediary interaction.

I now I gather you're (cough.) excuse me you're a visitor/3
U

I um yes are you part of the university or/5
U yes I am/4 well

I ya (,) um I I jus (,) we
U I teach at a Canadian university/6

I ask you this because i- its awful to bring up charges
U

I straight away (laugh) but just so that you know (,) you
U

I know that it's a ten pound basic an it's (inaud....)/7
U yes/8

I right ok (,) right (laugh...) we've got that out of the
U (laugh)

I way (laugh...) what's what's the subject of of y- your query/9
U (laugh)

I right (.....) any- anything
U greek turkish relations/10

I particularly (,) specific/11
U actually I'm interested in their

I right (,) disputes (...)
U (,) disputes other than Cyprus (....)/12

I (cough) (.....) other than Cy- are there any (,) any
U

I particular ones(...)/13 you know any/15
U um/14 the Aegean

I
U (,) dispute (.....) and the: their disputes over the

I
U treatment of (....) the (,) Turkish minority in Greece

Figure 2. Partial transcript of user-intermediary interaction

From these examples, we can see that the users and intermediaries engage in a number of activities that are seemingly unrelated to the specification of search topic, but which appear to be necessary to the eventual formation of an effective search strategy, and to efficient interaction between themselves and with the data base.

4. Desiderata for an intelligent computer interface

Presumably intelligent human intermediaries seem to perform a variety of functions in the information interaction, including such things as logging on to the system, choosing data bases, formulating a query and search strategy, applying system commands appropriately and choosing document displays. One or another of these functions has been the focus of most work in computer intermediaries or interfaces up to now. However, on the evidence above, there appear to be many other functions which intermediaries perform, which are perhaps more directly related to interaction with the user, and which have not been dealt with by most computer intermediary proposals. Since the admittedly rather scanty evidence to date seems to indicate that existing computer intermediaries are not terribly successful when used by end users, it may be of interest to consider what some of these other functions are, what roles they play, and how they relate to success of the interaction.

A truly intelligent computer interface would be one, then, which would perform all of the functions that an intelligent, successful human intermediary does, which are necessary and sufficient for successful information retrieval system performance. Furthermore, it would be necessary for these functions to be performed in an appropriate interactive dialogue, perhaps also based on that which is performed in human-human information interaction.

In the next section, we describe, rather briefly, a research program which has been aimed at developing an intelligent computer intermediary in just this sense; that is, one whose aims have been to investigate the functions performed in the information interaction, to determine which of these is necessary and sufficient for information interaction in general, and to specify the structure of the dialogue between intermediary and user. There are other obvious components of an intelligent intermediary which we take for granted as being necessary, such as natural language understanding and generation, but which we have not investigated, leaving them for others more competent in the task.

5. A research program on intelligent information interaction

We provide here a brief description of the methods and results of a number of projects, conducted at several sites, since about 1978, whose general goal has been the investigation of the information interaction, in order to specify the requirements for an intelligent intermediary. This work has been reported in, among others, the following publications. Belkin, Oddy & Brooks (1982); Brooks & Belkin (1983); Belkin, Seeger & Wersig (1983); Belkin, Hennings & Seeger (1984); Daniels, Brooks & Belkin (1985); Brooks (1986); Daniels (1986); Belkin & Kwasnik (1986).

The goals of these studies have been to:

- discover what functions are performed in human-human information interaction;

- discover which of these functions are necessary, possible and

desirable to simulate in human-computer interaction;
identify the resources necessary to perform these functions;
discover how the functions interact with one another;
discover what constitutes a good information interaction;
determine the structure of such an information interaction;
specify an architecture for such a system; and
investigate the retrieval techniques which such a system requires.

Our basic method for most of these investigations has been functional discourse analysis of real-life human-human information interaction, such as that represented in figures 1 and 2. This has meant the collection, on audiotape, of a large number of such interactions, their transcription, and their utterance-by-utterance analysis, according to the functions performed by the utterances, and the knowledge which was required in order to perform them. This basic technique has been supplemented by interviews, by simulation, and by direct investigation of possible retrieval techniques.

Our results in general include the following:

identification of a set of functions which we consider necessary and sufficient for at least the document retrieval situation (figure 3);

some investigation of function patterns in successful and unsuccessful interactions;

specification of some functions by subfunctions, and knowledge resources, sufficiently for implementation (although they are not yet coded);

specification of some function and subfunction interactions;

preliminary identification of a dialogue structure for document retrieval; and,

general specification of a system architecture for the information provision mechanism (figure 4).

It should be noted that most of the functions indicated in figure 4 are not those which have traditionally been ascribed to the intermediary, but are rather model building functions, associated with the intermediary constructing a representation of the user's goals, situation, problem and preferences, rather than directly specifying a search topic. These functions appear to be necessary, in order for the intermediary to proceed to the search topic formulation. Indeed, our results concerning patterns of functions seem to indicate that, although there is no single, serial or hierarchical order of functions that is appropriate to all interactions, the

model-building functions must be invoked at the beginning of the dialogue, in particular the problem description and user modelling functions, in order for the retrieval strategies function to be successful.

Name of function	Description
Problem state (PS)	Determine position of the user in the problem treatment process
Problem mode (PM)	Determine appropriate mechanism capability
User model (UM)	Generate description of user type, goals, knowledge, background, experience
Problem description (PD)	Generate description of problem type, structure, topic, context
Dialogue mode (DM)	Determine appropriate dialogue mode for specific situation
Retrieval strategy (RS)	Select terms, data base, build search strategy specify retrieval technique
Response Generator (RG)	Determine propositional structure of response, limits and constraints to response
Explanation (EX)	Describe mechanism capabilities, restrictions, administration, etc
Input analyst (IA)	Convert input from user into structures appropriate for other experts
Output generator (OG)	Convert instructions from DM and RG into appropriate formats for user

Figure 3. The functions of document retrieval information provision mechanism (After Belkin, Seeger & Wersig, 1983).

We have specified the User Model and Problem Description function sufficiently to identify the relevant subfunctions for each, and their knowledge resources in the particular document retrieval environment in which we worked, and also to specify and develop the knowledge representation formalisms which they will need; respectively, frames and partitioned semantic networks. Skipping over our other results, we will only mention here that the system architecture which our research has led us to adopt is, in general, the same as that proposed at this meeting by Croft & Thompson, and by Fox; that is, a distributed expert system, with blackboard communication.

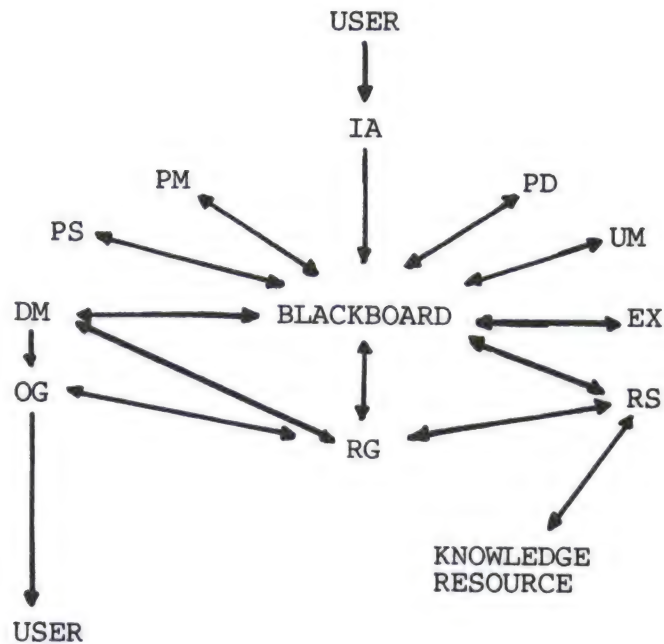


Figure 4. A distributed expert architecture for the information provision mechanism.

6. Conclusions

The conclusions which we draw from our analysis and results are that we ignore the model building and response generation functions of the information interaction at our peril, in the design of computer intermediaries, for it seems quite clear that mere 'user-friendliness', and just natural language understanding, without real problem understanding, are insufficient for the kind of interaction and models upon which a successful information system experience must be based. Selection of search terms and selection of data bases are not accomplished without an understanding of the user's situation, and it seems reasonably clear that we cannot shift the burden for successfully translating such understanding into terms appropriate for the IPM onto the user. Therefore, we suggest that an intelligent information system interface must at least incorporate these model building functions, and response generation functions, and must operate within a highly interactive, cooperative dialogue with the user. And furthermore, we suggest that we should be working toward such intelligent interfaces, if we intend to provide direct end-user access to information systems in the sense which was defined in the introduction.

7. References

BELKIN, N.J. & KWASNIK, B.H. (1986) Using structural

representations of anomalous states of knowledge for choosing document retrieval strategies. Paper presented at the ACM/BCS International Conference on Research and Development in Information Retrieval, Pisa, September 1986. In press.

BELKIN, N.J., HENNINGS, R.-D., & SEEGER, T. (1984) Simulation of a distributed expert-based information provision mechanism. Information Technology: Research, Development, Applications, 3:122-141.

BELKIN, N.J., ODDY, R.N. & BROOKS, H.M. (1982) ASK for information retrieval: parts I & II. Journal of Documentation, 38: 61-71; 145-164.

BELKIN, N.J., SEEGER, T. & WERSIG, G. Distributed expert problem treatment as a model for information system analysis and design. Journal of Information Science, 5: 153-167.

BROOKS, H.M. (1986) Developing and using problem descriptions. In: IRFIS 6: Intelligent information systems for the information society. Amsterdam, North-Holland: in press.

BROOKS, H.M. & BELKIN, N.J. (1983) Using discourse analysis for the design of information retrieval mechanisms. In; Research and development in information retrieval. Proceedings of the Sixth Annual International ACM SIGIR Conference, Washington, D.C., 1983. New York, ACM: 31-47.

DANIELS, P. (1986) The user modelling function of an intelligent interface for document retrieval systems. In: IRFIS 6: Intelligent information systems for the information society. Amsterdam, North-Holland: in press.

DANIELS, P.J., BROOKS, H.M. & BELKIN, N.J. (1985) Using problem structures for driving human-computer dialogues. In: RIAO 85. Actes of the conference: Recherche d'Informations Assistee par Ordinateur, Grenoble, France, March 1985. Grenoble, I.M.A.G.: 131-149.

TAYLOR, R.S. (1968) Question negotiation and information seeking in libraries. College and Research Libraries, 29: 178-189.

WERSIG, G. (1979) The problematic situation as a basic concept of information science in the framework of the social sciences: a reply to N. Belkin. In: Theoretical problems of informatics: new trends in informatics and its terminology. Moscow, VINITI: 48-57.

AUTHOR AFFILIATION: Nicholas Belkin is a Professor in the School of Communication, Information and Library Studies, Rutgers University.

ADDRESS: Rutgers University
4 Huntington Street
New Brunswick, NJ 08903

MACHINE INTELLIGENCE VS. MACHINE-AIDED INTELLIGENCE AS A BASIS FOR INTERFACE DESIGN

Linda C. Smith

The growing number of gateways and intermediary systems exhibit a variety of approaches to interface design, ranging from the machine-controlled "black box" which accomplishes as much as possible automatically to the human-controlled interactive dialog which makes available machine aids in search strategy development. Where machine intelligence dominates, an effort is made to keep as much control as possible within the computer by automating decision-making and execution of tasks. Where machine-aided intelligence dominates, the user is in control with the computer providing suggestions and gathering information to aid the user's decision-making. This paper explores the decisions involved in searching, the expertise required to make those decisions, and the appropriateness of machine intelligence vs. machine-aided intelligence as a basis for interface design.

Artificial Intelligence, Interface Design

1. INTRODUCTION

Prior to the availability of interactive computer systems, the emphasis in information retrieval (IR) research and development was on fully automatic IR. Since the 1960's, however, the technology has permitted the investigation of machine-aided information retrieval as an alternative, with designers seeking to use the capabilities of both human and machine. With the growing interest in gateways, front-ends, and intermediary systems, interface designers are exploring various approaches to dividing IR tasks between human and machine, accomplishing some automatically ("machine intelligence") and others semi-automatically ("machine-aided intelligence").

One can interpret the difference between machine-aided intelligence and machine intelligence as the distinction between augmentation and delegation. In augmentation the computer assists the user in the substance of the task; in delegation decisions are made by the computer using programmed criteria (1). Where machine-aided intelligence dominates, the user is in control with the computer providing suggestions and gathering information to aid the user's decision-making. Where machine intelligence dominates, an effort is made to keep as much control as possible within the computer by automating decision-making and execution of tasks.

IR can be thought of as problem solving, with the goal of locating information (in nonbibliographic databases) or bibliographic references (in bibliographic databases) relevant to the user's request. Ideally interfaces incorporating some elements of machine intelligence or machine-aided intelligence may prove helpful to both intermediaries and end users by leading to better solutions (more relevant and complete retrieval), faster solutions (more efficient retrieval), and/or by handling more complex problems. Approaches to making systems more intelligent can take two forms. Depending on how much is known about how expert human users accomplish a particular task, the simulation approach models the human user's techniques whereas the performance approach uses machine-based techniques which do not model human techniques but which still lead to improved performance. Another distinction worth noting is that between internal representations and external representations. When data to be used in search strategy formulation are to be handled automatically, one need be concerned only with internal representations, i.e., how they are to be represented for machine processing. However, if data are to be presented to the user to aid in decision making, then one must also be concerned with external representations, i.e., the displays of data at the user-computer interface.

2. DECISIONS IN ONLINE SEARCHING

To illustrate the complexity of online searching as a problem solving domain, it is helpful to review the decisions involved in search strategy formulation. Currently these decisions must be made by human users with little or no computer assistance. The decisions noted here are typical of those required in accessing bibliographic databases in an IR system using boolean logical operators, although many would also be required in searching in

other IR system environments as well.

Database selection. Selection of databases must take into account the attributes relevant to the request, considering such factors as subject coverage, time period coverage, document type coverage, searchable and printable fields, cost, and currency. The number of databases to be searched must be determined and, if more than one, the sequence must be decided. The choice of databases is likely to affect other decisions, since databases differ in their vocabulary and structure.

Concept identification. The concepts to be searched must be necessary and sufficient to represent the request and retrieve relevant items. The concepts which it is necessary to include explicitly in a search strategy formulation may vary with context. For example, a search on "behavioral effects of food additives" could be limited to the "food additives" concept in a social sciences database such as PsycINFO, but would have to include both concepts in databases which have articles discussing food additives in a variety of contexts.

Subject term selection. Possibilities for subject term selection depend on the database(s) to be searched. Terms may be drawn from controlled vocabularies and free text. Controlled vocabulary terms may include thesaurus terms and codes. Free text or natural language terms should reflect the full diversity of terminology which can appear in titles or abstracts. The task here is to identify "searchonyms"--terms equivalent for the purposes of searching (2)--which can include related terms, variant word forms and spellings, acronyms, and abbreviations. In addition one has the possibility of refining terms by using facilities such as truncation, word proximity (e.g., specifying that words be adjacent or a certain distance apart), and field designators (e.g., requiring that words be drawn from article titles). One must also avoid stopwords.

Selection of other types of terms or search criteria. Depending on the request and the database(s) to be searched, it may be necessary to select additional types of terms such as author names, language of publication, or time period. These must be designated using the conventions adopted by the particular database(s) to be searched.

Selection of logical operators. Search terms must be linked appropriately using NOT, OR, AND together with parentheses to insure proper sequence of execution. Where word proximity can be expressed, AND can be further refined by requiring that two words be in certain relative locations, such as in the same paragraph or adjacent.

Iterations. Recognizing that searching is a trial-and-error process, the user must decide whether and how to reformulate the initial search strategy formulation, determining 1) to broaden or to narrow and 2) by what means. The decision to reformulate is often prompted by sampling the retrieved set and using information gained from examples of relevant and nonrelevant documents as a basis for suggesting how to modify the strategy. Given the decision to broaden (seeking a larger retrieval set, higher recall) or to narrow (seeking a smaller retrieval set, higher precision), there are still many possibilities. Broadening can be accomplished

through truncating terms, adding "searchonyms", eliminating concepts, removing field designators, and loosening word proximity. Narrowing can include eliminating some searchonyms, adding additional concepts (with AND), adding field designators, using stricter word proximity, using limits (e.g., language), or eliminating some aspect using NOT.

3. EXPERTISE IN ONLINE SEARCHING

Given the complexity of the problem domain of online searching, it is evident that considerable expertise is required to enable good decisions to be made and the search to be conducted. There is not yet a detailed taxonomy of this expertise, although various writers on the role of the intermediary and the design of front-end systems have suggested categories which could be used to characterize this expertise. For example, Pollitt (3) lists four categories:

- 1) system knowledge--the command language and facilities available in the search system(s) from logging on and the submission of search statements to the printing of references or abstracts;
- 2) searching knowledge--relating to the strategy and tactics to be employed in searching;
- 3) subject knowledge;
- 4) user knowledge--knowledge about each individual user including previous searches and preferred journals.

To this list could be added 5) database knowledge--familiarity with the content and structure of available databases.

Other authors offer even more detailed listings. For example, Harter and Peters (4) present a typology of heuristics for online IR and Hamilton (5) provides a list of objectives for library instruction to enable users to be self-reliant and successful in accessing online systems. As noted earlier, one approach to making systems more intelligent involves simulating the expertise now possessed by human users. More research is required to determine in more detail the expertise underlying successful searching and the feasibility of embedding this expertise within the computer.

4. MACHINE INTELLIGENCE VS. MACHINE-AIDED INTELLIGENCE

Existing online IR systems are powerful tools providing access to rich information resources. However, as the enumeration of decisions and expertise given above illustrates, full exploitation of the capabilities of such systems is not a simple matter. The solution in the past has been to rely on highly trained intermediaries. But the environment is becoming sufficiently complex that both intermediaries and end users can benefit from some assistance. The choice confronting the interface designer is to determine what form this assistance should take: should the computer make decisions automatically (relying on machine intelligence) or should advice be provided to the user in making decisions (relying on machine-aided intelligence)? Available interfaces already implemented in gateways, front-ends, and intermediary

systems already demonstrate some of both approaches. Activities commonly automated include logon and translation of search strategies into appropriate command format. Machine aids offering advice are at a rather primitive stage of development. For example, advice on database selection often takes into account only subject coverage and not other factors such as currency and types of documents covered. Advice on selection of additional subject terms may be limited to suggesting terms which tend to appear in relevant documents already found.

The distinction between machine intelligence and machine-aided intelligence provides a convenient vehicle for evaluating available interfaces and for suggesting alternative designs. In assessing an interface to an online IR system, the following questions can be posed:

- 1) For which decisions in the search strategy formulation process does the interface offer assistance?
- 2) What form does this assistance take--is the decision made automatically by the computer (machine intelligence) or interactively with the computer offering advice to the user (machine-aided intelligence)?
- 3) What expertise underlies the assistance offered by the interface?
- 4) How well does the system model the expertise of a skilled human user? In what areas is it deficient?

Designers can rely on machine intelligence when a decision or procedure is straightforward and well understood. Otherwise, machine-aided intelligence is necessary to allow users to maintain some control of the outcome. Given the incomplete understanding of how skilled human users function, it is likely that collaboration between user and computer in IR will continue to be necessary for some time. Nevertheless, there are many possible forms which such collaboration can take. Available interfaces have not yet explored the full range of possibilities for machine intelligence and machine-aided intelligence.

5. REFERENCES

1. Paisley, W.; Butler, M. Computer Assistance in Information Work. Palo Alto, CA: Applied Communication Research, 1977. ERIC: ED 146900.
2. Attar, R.; Fraenkel, A.S. Local feedback in full-text retrieval systems. Journal of the Association for Computing Machinery 24:397-417, 1977.
3. Pollitt, A.S. An expert system as an online search intermediary. In: 5th International Online Information Meeting. Oxford: Learned Information, 1981, pp. 25-32.
4. Harter, Stephen P.; Peters, Anne Rogers. Heuristics for online information retrieval: a typology and preliminary listing. Online Review 9(5):407-424, October 1985.
5. Hamilton, Dennis. Library users and online systems: suggested objectives for library instruction. RQ 25(2):195-197, Winter 1985.

AUTHOR AFFILIATION: Linda Smith is Associate Professor in the Graduate School of Library and Information Science at the University of Illinois at Urbana-Champaign.

ADDRESS: University of Illinois at Urbana-Champaign
410 David Kinley Hall
1407 W. Gregory
Urbana, IL 61801

FUNCTIONALLY INTELLIGENT INTERFACE:
A COMPREHENSIVE MODEL OF HUMAN COGNITION INCORPORATING
BOTH POSSIBILISTIC AND PROBABILISTIC FUNCTIONING

Michael S. Pincus and Kathy W. Pincus

The major idea in developing man/machine computing tools to amplify intellect and accelerate the human learning process is pivotal on the balance of cognition and subcognition.

With our sub-cognitive tools, the emphasis is primarily on locating (retrieval) and manipulating (correlating) semantic cluster structures spread throughout a data or knowledge base. These programs utilize recursive iterative circuits which permit relational queries of data and are able to locate possibilistic joins. These connection networks, i.e. interconnections of abstraction in natural language, are points of relevance and concept convergence/divergence.

Through the use of our programs, information can be analyzed and abstractions studied rapidly in a micro environment through a mathematical process. The product for the human user is increased correlation of data which results in accelerated learning capability and analysis support toward subsequent decision-making.

As we continue to create new interfaces for man and computing machines, the machines themselves will by the sheer force of physics, model three dimensional connectivity. Therefore, it is really no longer workable for computer scientists to model computational systems in ways which are different than the universe in which we find ourselves.

Thunderstone is an AI software company specializing in data retrieval and correlation tools.

Interfaces, Natural Language Processing, Artificial Intelligence

THE FUNCTIONALLY INTELLIGENT INTERFACE:

To design a functionally intelligent interface between a human and a computer, one must start with a successful model upon which to extrapolate. For this model, let us use the manner in which experts deal with information in the solving of problems related to their field of expertise.

Traditional logic according to Dewey consists first of defining the nature of a problem before it can be solved. This is a key aspect of problem analysis, and must always be present before any decision-making can be undertaken. Where we have become accustomed to branched decision making as the subject of decision support computer software, the information analysis process has not generally been successfully aided by computer usage.

When experts analyze information, they utilize the relationships that exist between the main parts of a problem to conceptualize, or in a human sense to "parse" the main parts of the problem into smaller useful chunks. Experts recall aggregates of data (chunks or clusters) based on use, experience and functionality. This constitutes for most experts the ability to sense in a strange (previously unknown or unfamiliar) sub-cognitive way the structure of a problem before comprehending the details.

Thus we find an important parallel in the internal structure of a computer program which manipulates information, and in the mind of an expert who manipulates information.

As we find our attention drawn to this internal structure, it becomes of interest to inspect this relationship of the parts to themselves, to other parts, and to the whole. If we limit this model of problem solving to the assimilation and correlation of information, then we can view the parts as retrieved clusters of data, whether retrieved from the mind of an expert, or from the body of a resident data base in a computer. Larger parts (or the relationship of parts to parts) can be viewed as the combination of retrieved data clusters: in the mind of an expert this might be viewed as a string of concepts; in a computer this might be viewed as a small file made up of clusters of data, extracted from a larger data base. The whole, in the mind of an expert, might be considered to be his overall knowledge of a subject; in the computer this could be considered to be the totality of the data base you have available on the subject.

Putting our attention first on the parts, we must address ourselves to the clusters of information resident in a data base, or resident in someone's mind. Three qualities to these clusters are of interest to retrieval software design:

- A) What are the components to the cluster?
- B) What are the abstract qualities to the clusters?
- C) How does one integrate the clusters into larger engineered knowledge systems?

Some experts seem to deal with these clusters as independent elements, which are combined and recombined to represent different problem statements.

Solutions to problems using information retrieval methods consist of 3 major mental processes which come into play as part of the initial problem analysis stage.

- 1) Discerning the string of terms to use which define the problem (i.e., establishing the components which could make up a cluster).
- 2) Making both simple and complex comparisons of these strings to the text base in use (i.e., exploring the abstract qualities).
- 3) Storing in some way the resulting correlations for later use (for integration into knowledge systems).

This kind of 3-step process delineates the components to the cluster and can show the evolution of those data structures. Additionally the data cluster components are distinct units that the expert problem solver may consider useful by themselves. Operating in this way (i.e., separating and correlating the parts) reduces the load on an expert's working memory, thus paralleling the problem analysis process he must undergo in any case, and thus providing valid decision support for this stage of problem solving via a computer. This is what we mean by a functionally intelligent interface.

Additionally, expert problem solvers have been found usually to state a problem in terms of clusters of data containing abstractions of the information. These abstractions have the quality of having been derived from earlier data structures and are a reflection of the expert's knowledge of either the problem or the general area under consideration. Expertise in the manipulation of these abstractions represents an important element to data retrieval technology.

Expert problem solvers demonstrate sophisticated comprehension of the architecture of the data which represents their field or domain. They combine the simple elements of a problem into useful clusters. What many researchers and system designers have missed is that these useful clusters are then further organized into abstract structures for more creative use. This represents a model of learning which comes about as a result of the interaction of cognitive and sub-cognitive functioning.

COGNITIVE VS. SUB-COGNITIVE FUNCTIONING:

Decision support and information skill necessitate two levels of computer and human system function and compatibility (interface): cognitive and sub-cognitive. By cognitive is meant a conscious realization process by which decisions can be made, assumptions can be postulated, solutions can be invoked. This phase gets the most attention. However, cognition cannot come

about by itself; it is brought to awareness level as the result of sub-cognitive processes.

An expert involved in the analysis of a problem goes through a sifting and sorting process in which aspects of the problem (which one could also view as extractions of information from the whole of what is available to be considered) are inspected, contemplated, accepted, rejected, and reorganized. These are actually sub-cognitive (that is, just under, or prior to cognition) processes. Functional decision support by a computer would have to parallel this sub-cognitive process as well as the cognitive process to be deemed to be providing valid support for the entire analysis and decision-making process.

This necessity to interface cognitive with sub-cognitive functioning for both the human and the machine intelligence can be viewed in this way. First, in human terms, the sub-cognitive action of discovering some sort of pattern to information and how that information is structured sets up a model of the thought patterns which went into a certain data structure. This model can then be used cognitively to correlate and re-organize larger bodies of information. It is at this stage that actual decision-making becomes possible.

Secondly, in natural language processing terms, by isolating complex language relationships we are extracting a level of enfolded order distributed within a body of data. By doing this, we are able to discover the synchronicities and relationships which normally characterize complex linguistic structures.

These synchronicities and relationships at a gross level characterize the human cognitive functions which went into creating the macro info structures under analysis. At the macro level these functions are cognitive (known by the writer, intentional, organized), and at another level they represent the micro sub-cognitive (what came before, not necessarily known) level.

THE RELATIONSHIP OF COGNITIVE AND SUB-COGNITIVE FUNCTIONING TO THE HOLONOMIC MODEL OF COGNITION:

The human mind seems to represent perceptions holonomically. (with a view toward usage of the whole). That is, everything happening within the mind is enfolded into everything else and is distributed throughout the system. This holonomic paradigm models a certain facet of physical and mental functioning in that what people seem to do with their sense organs is use them to unfold the enfolded, or decode the coded (hidden, confused, or complex) order of things.

In the adequately functioning human, data extrapolation, association, and correlation is a naturally occurring activity. Holomovement of data can be defined as the useful conversion of a body of data into a model representing a state of energy or a "universe" of information. In a sense, a written or engineered

data base being the product of a human mind or group of minds can be likened to this holonomic model. Through various methods, the encoded order in this database can be discovered and utilized for the problem analysis and decision-making process.

Normally, a person does this to a certain extent when he reads something and makes sense out of it. Thus, valid decision support via a computer to be justified for use at all, must accomplish an accelerated "sense making" capability; or in other words, enhanced data assimilation skills must be brought about for the human as a routine result of interfacing a human with a computing machine, for such an interaction to be worthwhile.

BRINGING THE MODEL TO LIFE:

This can in fact be done. Through a series of unique data retrieval and correlation methods we can remodel portions of that "universe" into solidities which constitute a new representation of the data for human decision enhancement. Linear data goes from static to dynamic!

In other words, under the cognitive, naturally occurring and "logically" connected data structures also exists an encoded sub-cognitive implicit order which humans are not normally trained to discern, although we seem to intuit this level. If we could enhance this level of human data assimilation skill we could get much more out of the information we study.

Various mechanical computing processes can be useful in distilling these hidden ripples much like the way we can retrace the lines inside a hologram to the source. Fourier theory states that any pattern, no matter how complex, can be analyzed into regular wave forms which are called sine-waves. A decomposed or highly dispersed pattern can be recomposed or unified when sine-waves are synthesized or "convolved" with each other. This is the act of superposition or adding them one on top of the other.

Gabor, the inventor of holography added to the Fourier Theorem one more ingredient called the "ripple phenomenon", whereby each pebble in a pond leaves its own signature in the wave form and this can be analyzed and used to create 3D images. This is the basic idea behind holography. This type of function operates much the same way as do those functions which analysts use to perform statistical operations. They use FFT or Fast Fourier Transformations in order to speed the computation of mathematical correlations. We can also use these transformations to analyze both the cognitive representations and more particularly study sub-cognitive tracings inside of bodies of data of any sort. We could observe a body of information as if we were looking at a separate "universe".

It is accepted by physicists that beneath all physical matter is energy existing as waves. The longer the wave the weaker it is. The shorter, the more powerful. Where energy is

so reduced in its wave length that it is in a state of no movement (as a page in a book unread) it becomes infinite energy. Out of points of infinite energy other modes of energy resonate. Through mathematical interactions these slow and weak wave lengths create 3D forms and a physical world.

HOW CYBERNETICS COMES INTO PLAY:

Let us say that the shortest data/information structure we could have is a letter or symbol. The next level is a grouping of letter or symbols into vowels/consonants or expressions further, and further, into words or formulas. Then through various grammatical laws these things can be further grouped into sentence structures which result in paragraphs, pages, chapters, reports, files, etc. These structures can be manipulated and correlated with recourse to various data processing methods such as recursion which operates with data much like "feedback" operates cybernetically in nature.

These "feedback loops" are expressions whose output can be fed back into themselves as new input. An example of this might be the way a loudspeaker's sounds can cycle back into the mike and come out again; or how when playing racketball the sound of the ball being hit echos and re-resonates with its own sounds as it strikes the wall. It's a looping system where order slides smoothly into disorder and then reappears inside of other patterns. These are features that rely entirely upon the presence of feedback, and are unaware of anything else. AI scientists, data processing professionals, and mathematicians have begun studying the iteration of these type of looping systems using computers.

Recourse to these mathematical methods can position the deep structures in language and locate the logical joins and this task can be accomplished with relatively small software programs. According to Chomsky, deep structures in data, just like the genetic language that has come out of molecular mechanisms, have like elements that reflect the functional logic inherent in the operations of the central nervous system. There are parallels between the functional operations of DNA and RNA and the generative grammars of language. The 5 basic rules which establish this, are as follows:

1. Phonetic languages use different letters and symbols with different frequencies.
2. There are preferred sequences of letters and symbols.
3. There is general agreement on the definitions of words and various word and word combinations. Some are useful and others represent a certain degree of "noise".
4. Words have certain lengths.
5. Laws of pattern matching can be used to control the sequence, combination, and re-combination of words in sentences, paragraphs, pages, and files.

The basic idea of "feedback" modeled through recursion to correlate data is founded on one concept: the iteration of a real-valued mathematical function. The idea is that f can be representative of a word or string of symbols like a sentence, paragraph, or file as easily as it could represent any list of things. The interplay of such functions routinely incorporated in retrieval systems makes it possible for such systems to demonstrate a degree of "artificial" intelligence.

INTERFACING BOTH A POSSIBILISTIC AND A PROBABILISTIC APPROACH:

If we took data in any form and thought of it as a "universe", we could use string retrieval and recursion to further model it. Like a bird's eye view of cars moving down a multi-lane highway, or molecules of fluids moving inside of larger fluid jetstreams, what was once a 2 dimensional field of patterns of letters and numbers now becomes a multi-dimensional world of order and disorder interacting to generate new structures.

It is this order which represents probabilistic functioning, where the disorder represents possibility. Linear structures which can be read and follow a logical pattern represent probably logical patterns, i.e., the structure is predictable and follows a general train of thought. The non-linear and multi-dimensional deep inferential correlations represent the possibilities that exist at every turn. By being able to manipulate and inspect these possibilities, new creative patterns can be built.

In such a "universe" these concepts implemented through software could be used to study any new variety of correlation. Things will take on new unique states. Within this "universe" we can establish connection and interchange in discrete orbits by observing the patterning of patterns of meaning with some mutual frequency contingent on requisite considerations made by the user. In this way, possibilistic sub-cognitive bits aid the human in understanding and represent an acceleration of human perception and analysis. A patterning of the seemingly random fields of abstraction inside of a body of data produces structure. This constitutes a natural internal architecture which can be further controlled through feedback to decrease complexity. This is a process which occurs through the cooperation of a human user and a computer.

Manipulating these possibilities constitutes the thought process we call analysis and evaluation. It is the precursor to what comes after, that is, the probabilistic process of making a decision. The human who has this facility in a computer will be able to get more out of the information he reads, benefiting from an accelerated and enhanced analysis process, and will be better equipped for his decision-making phase to come.

For someone engaged in the serious effort of engineering a knowledge base into an expert system, this analysis stage is vital. One cannot engineer a probabilistic if-then branching

expert routine until the rules are clearly established. By use of a computerized possibilistic search and correlation tool, huge volumes of data can be perused, patterns established, and rules extracted. Possibilities are explored until probabilities are determined. These can then be engineered into the probable if-then branching structure provided with other kinds of tools.

CONCLUSION:

These concepts embodied in useable software/hardware technology constitute an innovative and comprehensive approach to data retrieval, analysis and correlation, and postulate a new level of man/machine interface. We anticipate that a design science for data retrieval and correlation following these models which carefully incorporate many naturally occurring phenomena is possible; its relevance is to the field of advanced data processing and AI: that is, "artificial" intelligence.

**** REFERENCES ****

1. Adelson, Beth: "Constructs and Phenomena Common to the Semantically-Rich Domains", Int'l. Journal of Intelligent Systems, John Wiley & Sons, Inc., 1986.
2. Bohm, David: Wholeness and the Implicit Order, Routledge & Kegan Paul, 1980.
3. M. Chi, R. Glaser, and E. Rees: "Expertise in Problem Solving," Advances in the Psychology of Human Intelligence, Lawrence Erlbaum, Hillsdale, NJ 1981, Vol. 1.
4. Chomsky, Noam: Language and the Mind, Harcourt, Brace, Jovanovich, 1972.
5. Hofstadter, Douglas: Metamagical Themas, Basic Books, 1985.
6. Jaynes, Julian: Origin of Consciousness in the Breakdown of the Bicameral Mind, Houghton Mifflin Company, 1976.
7. Pincus, M & K; Shafran, R.F.: "Microcomputer Expert Systems and CAI, Two Unique Approaches", Proceedings: Artificial Intelligence and Robotics Symposium (US Army TRADOC and US Army SSC), Sept 1984.
8. Pribram, Karl H., "Behaviourism Phenomenology and Holism in Psychology: A Scientific Analysis", Journal of Social and Biological Structures 2:65-72, 1979.
9. Shiffrin, R.M., and Schneider, W.: "Controlled and Automatic Human Information Processing: II. Perceptual Learning, Automatic Attending and a General Theory," Psychological Review, 84 127-190 (1977).

AUTHOR AFFILIATION: Michael Pincus is Chairman/CEO of
Thunderstone/EPI, Inc.
Kathy Pincus is President of Thunderstone/EPI, Inc.

ADDRESS: Thunderstone/EPI, Inc.
P.O. Box 839
Chesterland, OH 44026

AN OVERVIEW OF
THE I³R DOCUMENT RETRIEVAL SYSTEM

W.B. Croft
R.H. Thompson

A document retrieval system is described that performs similar activities to human intermediaries. The system improves the effectiveness of text retrieval by acquiring detailed models of users and their information needs and selecting search strategies based on these models. The system combines user specified domain knowledge and statistical techniques in order to provide efficient, effective retrieval. By using a single representation for both document and domain knowledge, browsing is facilitated.

User Modeling, Intelligent Intermediary, Artificial Intelligence,
Blackboard Architecture, Rule-Based System

An Overview of the I³R Document Retrieval System

W. B. Croft
R. H. Thompson

Abstract

A document retrieval system is described that performs similar activities to human intermediaries. The system improves the effectiveness of text retrieval by acquiring detailed models of users and their information needs and selecting search strategies based on these models. The system combines user-specified domain knowledge and statistical techniques in order to provide efficient, effective retrieval. By using a single representation for both document and domain knowledge, browsing is facilitated.

1 Introduction

Much research in information retrieval has concentrated on indexing techniques for representing the contents of documents and retrieval techniques that compare documents to queries [16, 21]. These techniques have supported the inference and uncertainty that is an essential part of document retrieval by a variety of methods. The most common method has been to base retrieval and indexing techniques on the statistical properties of the text. Probabilistic models of document retrieval are the best example of this approach [21]. Given a query and a set of documents represented by index terms derived from the text, the retrieval algorithms based on the probabilistic model infer a *probability of relevance* for individual documents and rank them accordingly. The probability of relevance is based on the frequencies of occurrences of index terms in the relevant and non-relevant sets of documents. The top ranked documents are shown to the user and the relevance judgments that are made provide information that enables the information need to be more precisely specified. This process of *relevance feedback* is one way of addressing the problem of ill-defined queries. The vector model [16] of document retrieval provides essentially the same form of inference. The advantages of using statistical techniques include the following,

- They are efficient to implement and can be integrated with a database system.
- They are more effective in terms of finding relevant documents than searches based on conventional Boolean retrieval techniques [15].
- They have a sound theoretical basis.

- They are independent of any particular domain. That is, different types of documents (journal articles, office memos) different domains (medicine, law) can be handled using the same techniques.

The major disadvantage of using these techniques is that the absolute performance is low. Although statistical techniques provide better performance than conventional Boolean techniques, there is still much room for improvement.

Given that the statistical techniques appear to have reached a ceiling of performance, researchers are investigating the application of techniques for natural language processing and knowledge representation [10, 18, 23]. There are, however, inherent difficulties in applying artificial intelligence (AI) techniques to the information retrieval task. In particular, the amount of domain-specific knowledge used even in a small collection of documents is very large. In many applications, the number of documents is also very large and dynamic. A natural way of limiting the amount of domain knowledge that must be considered is to concentrate on acquiring knowledge from individual users about specific queries. This domain knowledge can then be used to improve retrieval effectiveness by supporting inference based on the domain *concepts* (important words or phrases) in the queries and documents. The process of identifying concepts using domain knowledge can be seen as an extension of the process of identifying important index terms using limited statistical information. When domain knowledge is not available, the document retrieval system should be able to use well established statistical techniques to guarantee a reasonable level of performance.

In this paper, we describe a document retrieval system, I^3R , that combines the use of domain knowledge with statistical techniques. It provides a variety of facilities for query formulation, browsing, retrieval, and evaluation. I^3R is designed to act as an *expert intermediary* (hence the name *Intelligent Intermediary for Information Retrieval*). That is, the types of activities undertaken by the system during a search session are similar to those done by a human intermediary. The incorporation of the intermediary concept into an entire system design distinguishes I^3R from other "intermediary" system that provide access to conventional bibliographic systems [2, 11].

2 System Overview

A typical session with I^3R will involve three major phases; query formulation and refinement, search, and user evaluation. Document retrieval is a process of inference where documents whose contents can be inferred from the query are retrieved [22]. A retrieval system provides a particular form of inference, such as that used in the probabilistic model. The inference mechanism is limited in comparison to the inferences that would be made by the users of the system. By providing more information in the specification of the information need (the query), the system's limited inference mechanism will be able to retrieve more of the documents that would be retrieved by the users.

The I^3R system, during the query formulation and refinement phase, attempts to build an accurate description of the information need. This description is referred to as the *request model*. The request model contains, among other things, a list of concepts and their relative importance. These concepts are derived from the user's query and the domain knowledge. The domain knowledge itself may be acquired during this phase. The knowledge that is specific to an individual user is stored as part of a *user model*.

The information in the request model is used by the retrieval strategies to retrieve ranked lists of documents. When retrieval is viewed as an inference process, the ranking is produced by the certainty of the inference for each document. The current version of the system provides two statistical retrieval strategies. The main strategy is based on the probabilistic model and uses information about dependencies between terms and their relative importance derived from the concepts in the request model. For example, the request model may list *retrieval technique* as a concept with an associated importance of 0.7. This is used to specify that the words *retrieval* and *technique* should occur together in relevant documents. The

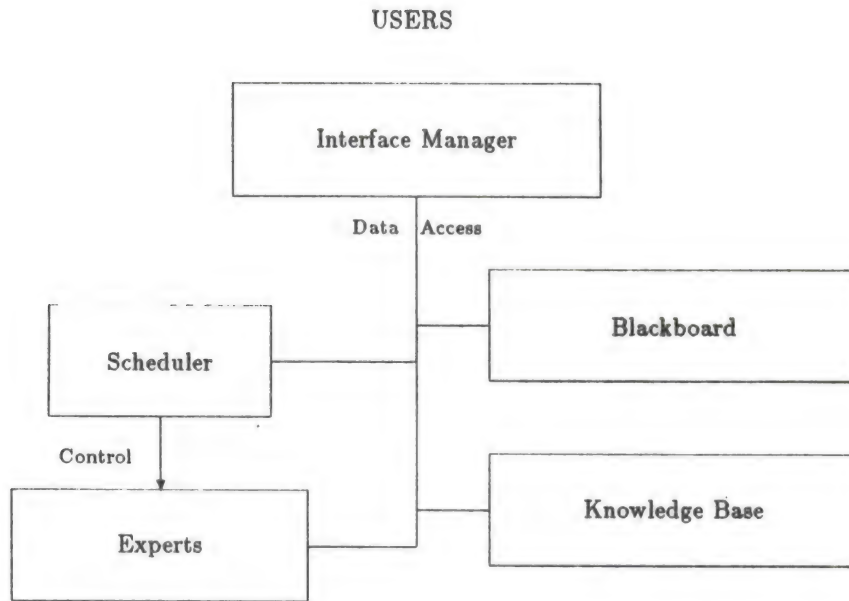


Figure 1: Major System Components

document is ranked according to the score [5, 6]:

$$\sum_i T(x_i) \cdot W(x_i) \cdot x_i + A$$

where x_i is the i th binary term in the set of terms describing a document, $T(x_i)$ is a weight related to the term's importance, $W(x_i)$ is a weight related to the collection frequency of term i , and A is a factor that depends on the presence of dependent groups of words in a particular document. The summation is done over all the terms in the query.

The evaluation phase involves the user looking at lists of relevant documents. The user is able to identify relevant documents and parts of documents such as phrases that are important. This information is used to update the request model. Evaluation is done in conjunction with the browsing facility. Browsing has been shown to be an important adjunct to the document retrieval process [13]. The I³R browsing facility provides access to the documents and other information in the database, along with commands to navigate along links to related information.

3 The System Architecture

The basic structure of the system (fig. 1) is derived from the Hearsay-II system [8]. This system is composed of a collection of independently operating knowledge sources (KS), each of which is an expert at solving one part of the problem. The activity of each of these experts is controlled by a *scheduler* which rates the value of proposed KS operations and chooses the one most appropriate for the current state of the problem solution. To determine an action to take, the KS's view a shared data structure called the *blackboard*. The structure

of the blackboard in some way represents the structure of the problem. It also provides an indirect means of communication and obviates the need for an expert to have any knowledge of the existence or operation of other experts.

3.1 Interface Manager

The Interface Manager (IM) is the component that controls the interface. Where the system's experts determine *what* information is to be displayed and collected, the IM determines *how* this is done. The fundamental elements of the interaction include multiple windows, a mouse, and a keyboard. With these elements the IM determines the layout of windows for document evaluation, for the graphical display of browsing paths, for the acquisition of domain knowledge, and others. The IM runs separately from the rest of the system and communicates with it by means of messages. Both the incoming and outgoing messages are posted on the system blackboard for examination by the system experts.

3.2 Blackboard

The Hearsay-II blackboard used different levels of abstraction to model the problem it solved. It is less clear that the text retrieval problem can be represented in this manner. Instead, the I³R blackboard is divided into a number of "areas" that represent different parts of the problem. These areas are:

- *Scheduler Information* – The current state of the scheduler plan and the *agenda* of rules to be executed during a the particular cycle.
- *User Model* – User stereotypes, expectations based on those stereotypes, and user supplied domain knowledge.
- *Request Model* – Documents, Terms, Concepts, and weights and dependencies associated with them.
- *Browsing Model* – Information related to what the browsing expert has sent to the Interface Manager to display and its previous recommendations.
- *I/O Requests* – Messages sent to and from the Interface Manager.
- *System Journal* – A list of rule invocations performed by the system.

3.3 Knowledge Base

The Knowledge Base is where the system maintains its permanent knowledge. It contains the text documents, information about the content of the documents, domain knowledge, and models of the users. The representation of this information using a database system is described in the next section. All of the system experts, the Interface Manager, and the scheduler have access to the information in the Knowledge Base.

3.4 Experts

In I³R the knowledge sources are called experts because they are implemented as small rule-based systems and, each one is devoted to handling A rule-based implementation was selected for two reasons. First, it allows incremental development. New capabilities can be added to the system relatively easily. Second, the explanation of the system's activity is facilitated by using a trace of rule invocations. The rules are condition/action pairs. The condition part is a predicate on the state of the system, which is represented by the blackboard. The predicate may consist of a number of clauses each of which tests a specific condition.

For example, a clause may test if a specific kind of message has been received from the Interface Manager or it may test how many searches have been conducted. The clauses are implicitly ANDed. The action part consists of the actions that are to be taken if the condition part is true. These actions may arbitrarily complex code. Some examples of rules are:

- A Search Controller Rule
IF
the last search was probabilistic AND no new relevant documents were retrieved
THEN
initiate a cluster search this time;
remove documents already judged;
send the results to the IM for user evaluation.
- A Request Model Builder Rule
IF
documents evaluations arrive from the interface manager
THEN
get the document evaluations from the message;
add the new evaluations to the document evaluation list;
get any term evaluations from the documents evaluations;
add the term evaluations to the term weights model.

In a typical blackboard system each of these rules would be a distinct knowledge source competing for system resources. Out of many rules eligible to fire only one would be able to. Alternatively, if the rules for an expert were combined into a large monolithic knowledge source, the activity of the system would be inexplicable to the user. In the present system there are six experts.

- The *User Model Builder (UMB)* collects information about the user to identify the stereotypes that apply. It accomplishes this by asking the user to estimate his competence in the subject area and in system use. If the user has used the system before, the expert will use this prior information to establish initial models for the current session.
- The *Request Model Builder (RMB)* constructs a model of the user's information need. The model consists of search terms extracted from the need statement and from the domain knowledge model of the user. Associated with these terms are a variety of weights which indicate such things as the frequency of a term in the document collection, in the need statement, and in documents judged relevant and not relevant.
- The *Domain Knowledge Expert (DKE)* uses the domain knowledge collected from the user and from experts in the field, to suggest additional concepts to enhance the user's request.
- The *Search Controller (SC)* selects and applies search techniques to retrieve documents likely to be relevant to the user's request.
- The *Browsing Expert (BE)* provides the user with an informal way of finding documents by allowing him to examine the contents of the database. The user can start from any document, author, or term he knows about and travel via links to other items. Using heuristics, the expert offers a suggestion of which path to follow to an item it considers to be useful. The user provides continuous evaluation of the browsing process.
- The *Explainer* enables the system to explain its actions and knowledge, as well as help the user with system use.

In summary, each expert has two basic activities. First, it can evaluate the state of the system and decide if there is something it should do. Second, it can perform an activity altering the blackboard. These activities among multiple experts can cause conflict. For example, the action of one expert can alter the blackboard, rendering the enabling conditions of a proposed activity of another expert invalid. For the effective operation of the system these activities must be coordinated.

3.5 Scheduler

This coordination is done by the scheduler. It does this by assigning priorities to the experts so that the most important actions happen first. The priorities are determined according to plans that the scheduler uses to determine the course of the session. The flow of the system's operation, as embodied by the scheduler plans, is to develop a detailed model of the user's information need, use this information for sophisticated searches based on probabilistic models, and get the user's evaluation of the system's performance.

These plans are organized as a hierarchy of goals, each of which must be satisfied in order to complete a retrieval session. The goals at the bottom of the hierarchy (leaves) are where the experts' priorities are defined. Only those that are appropriate for the particular phase of the session are listed at the associated leaf goal. For example, the Search Controller has no need to be active during the phase when the system is acquiring the initial description of the information need.

To represent these plans a transition network is used (fig. 2). Each state represents one of the goals. The states are connected by transitions which represent the possible plans steps. Associated with each transition are conditions which determine when the transition can be taken. The conditions examine the state of the blackboard and look at things such as the number of searches made and the number of relevant documents found. The transitions are of two kinds, normal and exceptional. The normal ones represent the standard or default sequence of steps that a retrieval session should take. The exceptional ones are taken when the system's performance is not proceeding as expected. For example, if, after five searches the system has only found 50 % of the expected number of relevant documents, an exception transition will be taken. The system would go into a state which gives priority to the experts concerned with developing the information need. This transition in the figure is from state 6 to state 3.

Given the structure of the scheduler and the experts, the basic operating cycle of the system proceeds as follows:

1. The scheduler determines state of the session plan.
 - a. The scheduler examines the blackboard and checks if a transition out of its current state can be taken.
 - b. If one can, then it takes it and enters a new goal state.
 - c. If it is not in a "leaf" state, it goes to step a. else it stops and the system proceeds to step 2.
2. The system checks for any incoming messages from the Interface Manager. If there are any, they are posted on the blackboard.
3. The experts appropriate to the state of the plan evaluate the state of the system to determine if there are any rules that are eligible to fire. Each expert chooses one from its eligible set and posts it on the agenda.
4. In priority order, the rules are executed, as long as the conditions which caused the action to be posted have not been invalidated by the actions of a higher priority rule.
5. Return to step 1.

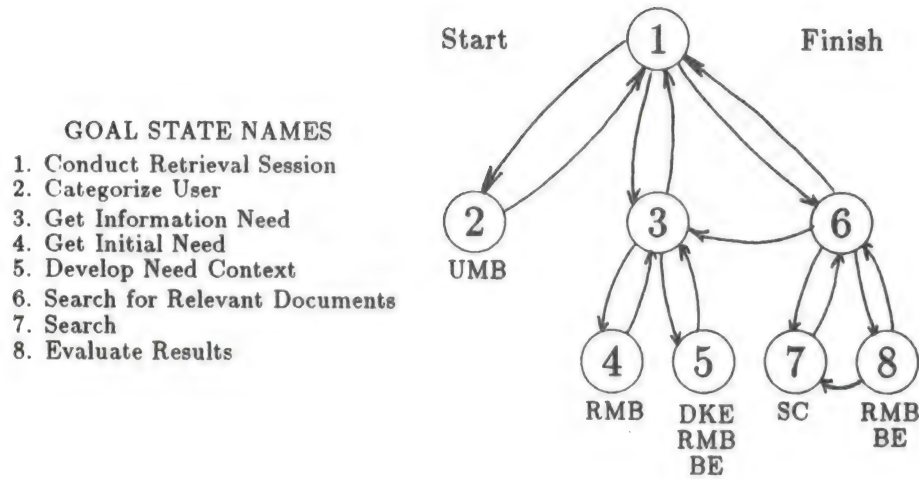


Figure 2: Scheduler States and Transitions

4 Knowledge representation

The knowledge in the I³R system can be put into four categories;

1. *Expert knowledge*: Rules in individual system experts
2. *Session-specific knowledge*: The Blackboard, including the current user and request models.
3. *Document database*: The document texts, their representatives, bibliographic information (e.g. author) and domain knowledge.
4. *User knowledge*: User models (including domain knowledge), user stereotypes, and search histories.

A relational database system is used for the documents and user knowledge. The other categories of knowledge are not readily represented as relations and are accessed through Lisp programs. Application programs are used to store and index new documents, as well as implementing the search strategies based on probabilistic models. The browsing expert and other experts access the database using the standard interface and can also be viewed as application programs.

In the categorization of the system knowledge, domain knowledge was mentioned as part of the knowledge base and the user knowledge. In general, this type of knowledge will be provided by individual users and will, effectively, be part of the model of that user. Different users will, in fact, have different views of the same area of interest and it is essential for I³R to capture these different views. However, if users are in the expert category for some domain, the knowledge they provide can become part of the general system "thesaurus" and available to all users. For convenience, we shall regard domain knowledge as part of the document database in the remainder of this discussion.

The document database contains a variety of information about the contents of the text documents. The simplest type of information is the documents themselves. Each document can be regarded as being made up of a text body, bibliographic header information (such as a title, authors and the source of the document),

and citations. This is obviously a simplified description of a document since in the office environment, for example, documents can contain other elements such as graphics and voice and can be of many different types, such as memos and reports. Recent research in this field has concentrated on the development of models in which complex document types can be represented [9, 24] and on the implementation of text data types for the long, variable-length streams of text found in documents [19]. In this paper, we shall assume we are dealing with the basic document which is relatively straightforward to represent in a conventional database system.

For effective document retrieval two other types of information about the documents are used by the I³R system. These are the *representatives* that summarize the content of the documents in a particular structure, and *domain knowledge* that provides some information about the concepts found in the text.

The document representatives in I³R are derived using statistical indexing procedures. A representative generated this way consists of a set of term numbers representing important word stems in the text and the frequencies of occurrence of those stems in the text. The other statistical information generated by the system is the strength of associations found in the set of terms and the set of documents. Essentially, each document and term is connected to its nearest neighbor as defined by a similarity measure [7]. These term-term and document-document relationships are used by search strategies based on different models.

The domain knowledge representation used by I³R was designed to satisfy the following criteria;

- It should be easy for a user to specify.
- It should be easy to update.
- It should be usable without sophisticated natural language processing techniques.
- It should be usable even when very incomplete.

The last two criteria emphasize the fact that in a system designed for text retrieval, the domain knowledge will always be incomplete and full understanding of a document text is not necessary for effective retrieval.

The representation used is similar to that developed for RUBRIC [20]. The user specifies pieces of semantic information through *rules* of different types and certainty values attached to those rules. These rules allow the user to describe synonyms, broader/narrower term hierarchies and how concepts are formed from single words. This is a superficial representation of domain knowledge compared to that used in, for example, the Conceptual Dependency work [1] but it provides the type of information that is useful for retrieval. An example of the use of this representation would be a particular user specifying the following rules that reflect a particular view of an area of interest;

- The words "relation" and "table" are strongly related.
- The words "relation" (or "relational") and "database" in the same piece of text form strong evidence that the concept "relational database system" is useful.
- "CODASYL", "Relational" and "Hierarchical" are all instances of the concept "Data Models".

The main goal of the system experts that use the domain knowledge is to establish the concepts that are related to a user's query. The certainty values are propagated to concepts that can be inferred from the words in the query and concepts with sufficiently high certainty are established. These concepts can then be used to enhance the request model and improve the search results.

A single abstract representation is used for the entire knowledge base. All the information in this knowledge can be described in terms of a network of nodes and links [7]. The nodes represent documents and concepts. The concepts include the word stems identified by the indexing procedures and the concepts

described with domain knowledge rules. The links represent both the statistical and semantic information about document content. In each case, some information such as a certainty value or weight is associated with the link. The following types of links can be described;

- Document-Concept: These links primarily describe the representatives formed by the indexing process. This information takes the place of the inverted and sequential file organizations used in conventional document retrieval systems and is the basis of the statistical retrieval strategies. There is also the possibility of users associating more sophisticated concepts (such as those described with rules) directly with particular documents. It should be noted, however, that this form of indexing has not proved effective in previous experiments.
- Document-Document: These links represent statistical associations derived from the document representatives or citations. Both types of information can be used in search strategies.
- Concept-Concept: This type of link represents both the statistical associations between single word concepts and the rules that form the thesaurus or domain knowledge.

The representation of the Knowledge Base as a network has a number of advantages. First, this representation is particularly suitable for implementation with a relational database system. Croft and Parenty [7] describe how statistical search strategies can be implemented using a network stored in a database system. The use of the domain knowledge is also simplified in that the system expert responsible for establishing concepts consists of strategies for using the information in the database (both statistical and semantic) and does not include specific domain knowledge. The propagation of certainty values is readily implemented using a network representation of the rules. Another important advantage is that the network makes both statistical and semantic information available for browsing by the users. As mentioned earlier, browsing is a critical part of the I³R system and the expert responsible for this aspect of the system consists of a set of rules for navigating through the network.

The other part of the I³R knowledge that is stored in the database system is the user knowledge. This consists of user models, search histories and user stereotypes. The user models include information about user preferences (e.g. interaction style) as well as domain knowledge. The search histories are summaries of the information built up in the Short Term Memory during particular sessions. User stereotypes provide prepackaged information about general categories of users. Examples of stereotypes are users interested in recall or precision-oriented searches, users with no system experience, and users with a high level of domain knowledge.

5 Conclusion

The I³R system represents an innovative approach to improving the usability and effectiveness of text retrieval systems. Its architecture provides a framework which allows the system to bring to bear many kinds of knowledge to assist the user in solving his information problem. Furthermore, the architecture allows the system to be extended the inclusion of new "experts".

6 References

1. Barr, A. and Feigenbaum, E. A. *The Handbook of Artificial Intelligence*, Wm. Kaufmann Inc, Los Altos, Calif., Vol. 1, 1981, 300-305.
2. Brajnik, G., Guida, G., Tasso, C. An expert interface for effective man-machine interaction. in

Cooperative Interactive Systems, Ed. L. Bolc. Springer-Verlag; 1985.

3. Brooks, H. M., Daniels, P. J., Belkin, N. J. Using problem structures for driving human-computer dialogues. *Proc. of RIAO-85, Recherche d'Informations Assistée par Ordinateur*, Grenoble, March 1985, 645-660.
4. Croft, W. B. Implementing a text storage and retrieval tool for the office. *Proc. IEEE First International Conference on Office Automation* (1984), 137-144.
5. Croft, W.B. Boolean queries and term dependencies in probabilistic retrieval models. *Journal of the American Society for Information Science*. **37**:71-77; 1986.
6. Croft, W.B. User-specified domain knowledge for document retrieval. *Proceedings of the International Conference on Research and Development in Information Retrieval*, Pisa (to appear).
7. Croft, W. B. and Parenty, T. J. A comparison of a network structure and a database system used for document retrieval. *Information Systems*, (to appear).
8. Erman, et al. The Hearsay-II speech-understanding system: integrating knowledge to resolve uncertainty. *ACM Computing Surveys*, **12**, (1980), 213-253.
9. Gibbs, S. and Tsichritzis, D. A Data Modeling Approach for Office Information Systems. *ACM Transactions on Office Information Systems*, **1**, 4 (October 1983), 299-319.
10. Lebowitz, M. Intelligent Information Systems. *Proc of the Sixth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1983, 25-30.
11. Marcus, R.S. An automated expert assistant for information retrieval in the information community, *Proceedings of the 44th ASIS annual Meeting*. 1981, 270-273.
12. McLeod, I. A. and Crawford, R. G. Document retrieval as a database application. *Information Technology: Research and Development*. **2**, 1 (January 1983), 43-60.
13. Oddy, R.N. Information retrieval through man-machine dialogue. *Journal of Documentation*. **33**:1-14; 1977.
14. Rich, E. Building and Exploiting User Models. *Tech. Report CMU-CS-79-119*, Carnegie-Mellon Univ., 1979.
15. Salton, G., Fox, E., Wu, H. Extended Boolean Information Retrieval. *Communications of the ACM*, **26**, 11 (Nov. 1983), 1022-1036.
16. Salton, G. and McGill, M. *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, 1983.
17. Schek, H. J. and Pistor, P. Data structures for an integrated data base management and information retrieval system. *Proceedings of the Eighth International Conference on Very Large Data Bases* (1982), 197-207.

18. Spark Jones, K. and Tait, J. I. Automatic Search Term Variant Generation. *Journal of Documentation*, 40, 1 (March 1984), 50-66.
19. Stonebraker, M., et al. Document Processing in a Relational Database system. *ACM Transactions on Office Information Systems*, 1, 2 (April 1983), 143-158.
20. Tong, R. M. RUBRIC - An Environment for Full Text Information Retrieval. *Proc. of the Eighth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (1985), 243-251.
21. van Rijsbergen, C.J. *Information Retrieval*, Second Edition, Butterworths, London, 1979.
22. van Rijsbergen, C.J. A non-classical logic for information retrieval. *Proceedings of the International Conference on Research and Development in Information Retrieval*, Pisa (to appear).
23. Walker, D. E. Natural-language-access systems and the organization and use of Information. *COLING-82, Proc. of the Ninth International Conference on Computational Linguistics*, North-Holland, Amsterdam, 1982, 407-412.
24. Zdonick, S. B. Object Management System Concepts. *Proc. of the Second ACM-SIGOA Conference on Office Information Systems*, SIGOA Newsletter, 5, (1984), 13-19.

AUTHOR AFFILIATION: Roger Thompson is a Research Assistant at the University of Massachusetts, Amherst, MA

Bruce Croft is an Associate Professor at the University of Massachusetts, Amherst, MA

ADDRESS: COINS Department
Room 305
Graduate Research Center
University of Massachusetts
Amherst, MA 01003

A DESIGN FOR INTELLIGENT RETRIEVAL: THE CODER SYSTEM

Edward A. Fox

Information retrieval systems could be dramatically improved if they would more effectively analyze, index, represent, search, match, rank, and present documents to users who should at the same time be better aided and understood. One partial solution uses an enhanced version of the SMART retrieval system which supports extended Boolean or natural language queries and several feedback techniques. With the help of a front-end system developed to interact with the menu-oriented Virginia Tech Library System (VTLS) a large scale experiment is underway with VTLS and SMART.

A more complete solution is also being sought with Composite Document Expert/extended/effective Retrieval, which employs distributed processing; logic programming; blackboards surrounded by a community of experts; and a large, relational, English lexicon to provide an integrated intelligent information retrieval system. CODER is being developed in Prolog and C as a test bed for prototyping and validating AI approaches for creating a knowledge representation from document collections, and for interacting closely with users who express their needs using natural language or more structured notations, as well as by making judgments regarding terms and document passages that have been presented using any one of several interfaces.

Blackboard Architecture, Document Knowledge Representation, Expert System, Extended Boolean Queries, Feedback, Menus, Natural Language Processing, Prolog, Relational Lexicon

INTRODUCTION

A revolution in the use of computers for accessing information is gathering momentum in the 1980s due to advances in microprocessors, telecommunications, and storage media [FOX 86a]. This presents an exciting opportunity to researchers and developers of information retrieval systems who now have adequate hardware support to be able to employ advanced methods for improved performance and at the same time to make these systems easy for people to use. A new generation of intelligent intermediaries is now being born!

At Virginia Tech a number of experimental investigations are underway to determine how best to apply recently developed information retrieval (IR) methods. One effort involves employing advanced IR techniques on a large online public access catalog. The Virginia Tech Library System (of VTLS, Inc.) is a menu-oriented library automation system used at Virginia Tech and around the world; a front-end system, ELSI (Enhanced Library System Interface) is nearly complete [FOX 86b]. It will allow a command-language style of interaction that many users prefer, and will become a vehicle for experimentation with query handling and ranking algorithms that can be performed on a personal microcomputer front-end (similar to that described in [MORR 83]).

Other studies will take place after 100,000 or more entries from the Virginia Tech library catalog database are loaded into an enhanced version of the SMART system [SHAR 86]. SMART has long served as an experimental system to validate the utility of new IR techniques [SALT 80]. Early studies indicated that its fully automatic indexing and retrieval performed no worse than a well-known retrieval system where trained indexers and searchers employed a special controlled vocabulary [SALT 72]. SMART was redone in the C language for the UNIX environment [FOX 83b] and has been extended to work efficiently for multiple online users [BUCK 85]. The vector space model [SALT 75], which typically applies to natural language queries, has been extended for SMART to allow manipulation of a variety of other concept types such as citations, cocitations, and bibliographic coupling counts [FOX 83a]. Searches using all of this data can retrieve more relevant documents than if only one type is employed [FOX 83c]. An enhanced form of Boolean logic is also accepted, and leads to much more effective retrieval [SALT 83b], especially when feedback queries are automatically prepared by the system [SALT 85]. The current version of SMART is being revised to not only include all these features; but also to have an interface like ELSI, and to run with very large collections of documents.

Another line of research, which is the focus of the rest of this report, deals with the CQomposite Document Expert / extended / effective Retrieval (CODER) system. CODER was originally proposed to support a wide range of experimental studies of composite documents [FOX 85]. CODER has been designed [FOX 86c] so that it can be used to test a variety of techniques employed by

artificial intelligence (AI) researchers, which seem especially useful for analyzing and retrieving passages [OCON 80] or whole documents. CODER has a flexible architecture that enables addition or change of any one of many different modules [FOX 86d]. Initial testing will be with a collection of ARPANET AIList electronic mail digests, where the type of each message must be determined to help with the analysis of document structure [FOX 86e].

Experimentation with CODER will be aimed at finding answers to such important issues of information retrieval (IR) research as:

- 1) Can an intelligent automatic intermediary serve users as well or better than a human intermediary.
- 2) Can knowledge of user characteristics [BORG 84] and a new style of graphics-based interface [FREI 83], where problem formulation, query construction, term expansion, feedback, browsing, and profile-based filtering are all interwoven in a highly interactive human computer dialog [ODDY 77], lead to more effective and pleasant retrieval, or to a new style of use of books and encyclopedias [WEYE 84].
- 3) Have natural language processing and knowledge representation methods advanced sufficiently so that more comprehensive document representations can be efficiently produced for heterogeneous collections.
- 4) Can findings about document structure resulting from studies of office information systems [PEEL 85] be applied to document analysis and indexing tasks.
- 5) Can several machine readable dictionaries be analyzed and utilized to aid in document and query processing.
- 6) Since retrieval according to different methods tends to lead to location of largely non-overlapping sets of relevant documents [KATZ 82], can rules be developed [HAYE 85] for integrating use of a variety of storage structures and search strategies, so that near-optimal choices are usually made.
- 7) Are the logic programming [KOWA 79] paradigm in general, and the Prolog language in particular, appropriate for developing an experimental IR system [SUBR 85].
- 8) Can expert system methods [HAYE 83], enhanced by the presence of a blackboard and strategist, be effectively used in an IR system (see [BELK 84] for motivation).
- 9) In the environment of full text databases [TEN 84], will an intelligent IR system be able to achieve higher levels of recall, for reasonable precision, than is now the case with commercially available systems [BLAI 85].

RELATED WORK

Interest in applying AI methods to IR problems has primarily focused on improving query formulation (ex., RUBRIC, where days may be spent to supply domain knowledge about a query [TONG 83]). Natural language work in this regard has aimed at better

understanding and expanding a query [SPAR 84]. More structured queries can also be handled when knowledge representation tools such as frames [MINS 75] are carefully adapted [PATE 84].

In a particular query, terms originally presented can be supplemented with "searchonyms" [ATTA 77] that are equivalent for search in the current collection only. Since query expansion using linguistically related terms can improve retrieval effectiveness [FOX 86f], presumably for many collections, there has been interest in automatically cataloging lexical and semantic relationships [EVEN 85]. Early work in lexicology [APRE 69] provided some of the motivation for specifying what is desired for a complete lexicon [EVEN 79]. Large projects in natural language processing have had to devote time to lexicon construction [WHIT 83], but with machine readable dictionaries [AMSL 84] there is hope for building a version that could be readily usable by others [AHL 85]. Given a dictionary, semantic hierarchies can be identified [CHOD 85], and it appears that the definitions themselves can be parsed to release further information [AHL 83].

Document processing has been considered by a few researchers. In the TOPIC system, a topical outline of the content of a document is produced with the aid of a word-expert parser [HAHN 84]. Work with Prolog for natural language parsing has been of interest for a number of years [PERE 83]. Representation of the content of documents is also essential; frames [FIKE 85] have been viewed by many as appropriate.

Since document analysis and retrieval are such complex problems, where knowledge from many sources must be integrated, use of a blackboard [ERMA 80] has been recommended, and has been applied to the I³R system [THOM 83]. When combined with a message-passing model, where transactions are handled as atomic actions by the blackboard, efficient control is possible [ENSO 85].

Prolog has been suggested as an appropriate language for use in building complex expert systems [HELM 85]. There are, however, many limitations of Prolog [BOBR 85]. MU-Prolog does provide significant advantages over other dialects, though, since a high efficiency database capability has been added to the interpreter [NAIS 84].

CODER FUNCTION AND STRUCTURE

CODER has been designed to serve both as a prototype for a new style of intelligent information retrieval, and as an experimental test bed that can be employed to answer some of the questions given above. As can be seen from Figure 1, it should be adaptable to a wide variety of sources of input and a diverse community of users. It should operate on one or more computers which run the UNIX operating system and communicate using TCP/IP (protocols employed in many current local area and long-haul networks). Multiple sessions of document and query processing can thus take

place in this distributed AI system (see Figure 2).

The overall structure of CODER is shown in Figure 3. The basic objects include: blackboards, experts, external knowledge bases, and resource managers. While the blackboards as well as the experts are controlled by strategists, the bulk of the processing is done by the various experts. These communicate with each other by way of the blackboard, but can communicate directly with external knowledge bases (eg., the document collection) and with resource managers (ex., the user interface). An example of this can be seen in Figure 4, which focuses on the user interface portion of the retrieval subsystem. The flow of control can be best understood by considering Figure 5.

While the design of CODER bears some similarity to that of I³R, there are many notable differences. First, it addresses the overall IR problem, not just the question of improving retrieval. Second, it has a finer granularity, so that experimental studies can be simplified by only requiring changes in a few specialized modules. Third, it has been designed from the ground up to operate in a distributed processing environment, using logic programming methods, and to support extensive natural language processing. Finally, there is a strong focus on composite document handling, where the structure of a document must needs be described using special knowledge representation techniques such as frames and relations.

Key to the goal of investigating the value of natural language processing of documents and queries is the CODER lexicon. While one part of the lexicon deals with domain knowledge (e.g., facts obtained automatically from the machine readable form of the *Handbook of Artificial Intelligence*), the second part is domain independent.

CODER LEXICON

The largest part of the lexicon is derived from machine readable dictionaries. More than 80,000 headwords have been processed from one large dictionary (CED - the *Collins English Dictionary* [HANK 79]) and are stored in the form of Prolog facts [WOHL 86]. A second dictionary [HORN 74] has been partially processed by another researcher [MITT 85] and after further local handling will also be added. Two smaller dictionaries [COWI 75,83] for handling verb and particle combinations, and a variety of idioms and phrases, are now being worked on and will eventually be included in the lexicon as well.

Figure 6 shows the levels of knowledge present in the CED, and Figure 7 describes the initial set of Prolog facts produced from the original typeset form. Work is underway to refine a grammar recently developed for analyzing adverb definitions, and will be followed with similar efforts for other parts of speech so that maximum benefit can be obtained from this type of relational lexicon.

In Figure 8 a typical document and related query are shown. Retrieving this document, which is clearly relevant to the question, is problematic, largely due to lexical-semantic problems. The only non-trivial word in the query that is also in the document is "chess." Only synonyms or related words appear for query terms such as "program," "defeat," "human," and "opponent." However, by appealing to facts taken from the CED knowledge base that are shown in Figure 9, a very convincing case for selecting this document can be made. Clearly, knowledge in the CED can be applied to aid in the retrieval process.

CONCLUSION

Research at Virginia Tech has focussed on studies relating to the use of advanced IR methods. On the one hand, a large scale validation of techniques previously tested with the SMART system is underway, building upon work with a front-end system specially developed to operate with the Virginia Tech library (using VTLS). On the other hand, CODER is an investigation of the applicability of AI methods for further enhancing the capabilities of IR systems.

CODER will serve as a test bed for experimental studies in this area, so that scientific knowledge can be obtained about the behavior and interaction among the various parts of the indexing, search and retrieval processes. Ultimately, it is hoped that methods which are shown to be both practical and effective will find their way into an upcoming generation of intelligent intermediary systems.

BIBLIOGRAPHY

- [AHL 83] Ahlswede, T.E. A Linguistic String Grammar of Adjective Definitions from Webster's Seventh Collegiate Dictionary. In *Humans and Machines*, S. Williams (ed.), 101-127, 1983.
- [AHL 85] Ahlswede, T.E. A Tool Kit for Lexicon Building. In *Proc. of the 23rd Annual Meeting of the ACL*, 268-276, July 1985.
- [AMSL 84] Amsler, R.A. Machine-Readable Dictionaries. *ARIST*, 19:161-209, 1984.
- [APRE 69] Apresyan, Y.D., I.A. Mel'cuk, and A.K. Zolkovsky. Semantics and Lexicography: Towards a New Type of Unilingual Dictionary, In *Studies in Syntax and Semantics*, ed. F. Kiefer, 1-33. Dordrecht - Holland: D. Reidel, 1969.
- [ATTA 77] Attar, R. and Aviezri S. Fraenkel. Local Feedback in Full-Text Retrieval Systems. *J. ACM*, 24(3): 397-417, July 1977.
- [BELK 84] Belkin, N.J., Hennings, R.D., and T. Seeger. Simulation of a Distributed Expert-Based Information Provision Mechanism. In *Inf. Tech.: Res. Dev. Applications*. 3(3): 122-141, 1984.

- [BLAI 85] Blair, D.C. and M.E. Maron. An Evaluation of Retrieval Effectiveness for a Full-Text Document-Retrieval System. *Commun. ACM*, 28(3):289-299, March 1985.
- [BOBR 85] Bobrow, D.G. If Prolog is the Answer, What is the Question? or What it Takes to Support AI Programming Paradigms. In *IEEE Trans. on Software Engineering*, 11(11): 1401-1408, November 1985.
- [BORG 84] Borgman, Christine L. Psychological Research in Human-Computer Interaction. *ARIST*, 19:33-64, 1984.
- [BUCK 85] Buckley, C. Implementation of the SMART Information Retrieval System. TR 85-686, Cornell Univ., Dept. of Comp. Sci., May 1985.
- [CHOD 85] Chodorow, Martin S., Roy J. Byrd, and George E. Heidorn. Extracting Semantic Hierarchies from a Large On-Line Dictionary. In *Proc. of the 23rd Annual Meeting of the ACL*, 299-304, July 1985.
- [COWI 75] Cowie, A.P. and R. Mackin. *Oxford Dictionary of Current Idiomatic English. Volume 1: Verbs with Prepositions & Particles.* Oxford Univ. Press, Oxford, 1975.
- [COWI 83] Cowie, A.P., R. Mackin, and I.R. McCaig. *Oxford Dictionary of Current Idiomatic English. Volume 2: Phrase, Clause & Sentence Idioms.* Oxford Univ. Press, Oxford, 1983.
- [ENSO 85] Ensor, R.J., and J.D. Gabbe. Transactional Blackboards. *Proc. of IJCAI '85*, 340-344, August 1985.
- [ERMA 80] Erman, L.D., Hayes-Roth, F., Lesser, V.R., and D.R. Reddy. The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty. *ACM Comp. Surveys*, 12:213-253, 1980.
- [EVEN 79] Evens, M.W. and R.N. Smith. A Lexicon for a Computer Question-Answering System. *Am. J. Comp. Ling.*, Microfiche 83, 1979.
- [EVEN 85] Evens, Martha., J.Vandendorpe, and Yih-Chen Wang, Lexical Semantic Relations in Information Retrieval. In *Humans and Machines: The Interface Through Language*, Ablex, ed. S. Williams, 1985.
- [FIKE 85] Fikes, Richard and Tom Kehler. The Role of Frame-Based Representation in Reasoning. *Commun. ACM*, 28(9):904-920, Sept. 1985.
- [FOX 83a] Fox, E.A. Extending the Boolean and Vector Space Models of Information Retrieval with P-Norm Queries and Multiple Concept Types. Dissertation, Cornell University, University Microfilms Int., Ann Arbor MI, Aug. 1983.
- [FOX 83b] Fox, E.A. Some Considerations for Implementing the SMART Information Retrieval System under UNIX. TR 83-560, Cornell Univ., Dept. of Comp. Sci., Sept. 1983.
- [FOX 83c] Fox, E.A. Combining Information in an Extended Automatic Information Retrieval System for Agriculture. In *The Infrastructure of an Information Society*, ed. B. El-Hadidy and E.E. Horne, North-Holland, Amsterdam, 449-466, 1984.
- [FOX 85] Fox, E.A. Composite Document Extended Retrieval: An Overview. In *Res. & Dev. in Inf. Ret., Eighth Annual Int. ACM SIGIR Conf.*, Montreal, 42-53, June 1985.

- [FOX 86a] Fox, E.A. Information Retrieval: Research into New Capabilities. In *CD-ROM: The New Papyrus*, Steve Lambert and Suzanne Ropiequet (eds.), Microsoft Press, 1986, 143-174.
- [FOX 86b] Fox, E.A. and S. Birch. UNIX Micros for Students Majoring in Computer Science and Personal Information Retrieval. *Microcomputers for Information Management*, 3(1):15-29, March 1986.
- [FOX 86c] Fox, E. A. and R. K. France. A Knowledge-Based System for Composite Document Analysis and Retrieval: Design Issues in the CODER Project, TR-86-6, Virginia Tech Dept. of Comp. Sci., Blacksburg, VA, March 1986.
- [FOX 86d] Fox, E. A. and R. K. France. Architecture of a Distributed Expert System for Composite Document Entry, Analysis, Representation, and Retrieval. *Proceedings Third Annual USC Comp. Sci. Symp.; Knowledge-Based Systems: Theory and Applications*, March 31-April 1, 1986, Columbia, S.C.
- [FOX 86e] Fox, E. A. Expert Retrieval for Computer Message Systems, TR-86-13, Virginia Tech Dept. of Comp. Sci., Blacksburg, VA, June 1986.
- [FOX 86f] Fox, E.A. Improved Retrieval Using a Relational Thesaurus Expansion of Boolean Logic Queries. In *Relational Models of the Lexicon*, Martha Evens (ed.), Cambridge Univ. Press, (to appear).
- [FREI 83] Frei, H.P. and Jauslin, J.F. Graphical Presentation of Information and Services: A User Oriented Interface. *Inf. Tech.: Res. Dev.*, 2(1):23-42, Jan. 1983.
- [HAHN 84] Hahn, U. and Reimer, U. Heuristic Text Parsing in 'Topic': Methodological Issues in a Knowledge-based Text Condensation System. In *Representation and Exchange of Knowledge as a Basis of Information Processes*, ed. by Hans J. Dietschmann, North-Holland, New York, 143-163, 1984.
- [HANK 79] Hanks, P. ed. *Collins Dictionary of the English Language*, William Collins Sons & Co., London, 1979.
- [HAYE 83] Hayes-Roth, F., Waterman, D.A. and Lenat, D.B., eds. *Building Expert Systems*, Addison-Wesley, Reading, MA, 1983.
- [HAYE 85] Hayes-Roth, F. Rule-Based Systems. *Commun. ACM*, 28(9):921-932, Sept. 1985.
- [HELM 85] Helm, A.R., Marriott, Kimbal, and Catherine Lassez. Prolog for Expert Systems: An Evaluation. *Proc. of Expert Systems in Government Symp.*, 284-293, October 1985.
- [HORN 74] Hornby, A.S. ed. *Oxford Advanced Dictionary of Current English*, Oxford University Press, Oxford, 1974.
- [KATZ 82] Katzer, J., et. al. A Study of the Overlap Among Document Representations. *Syracuse Univ. Sch. of Info. Studies*, 1982.
- [KOWA 79] Kowalski, R.A. *Logic for Problem Solving*. Elsevier North-Holland, New York, 1979.
- [MINS 75] Minsky, M. A Framework for Representing Knowledge. In *The Psychology of Computer Vision*, ed. by P. Winston, McGraw-Hill, New York, 1975.
- [MITT 85] Mitton, Roger. A Description of the File OALD.DAT. Personal Communication, July 18, 1985.

- [MORR 83] Morrissey, J. An Intelligent Terminal for Implementing Relevance Feedback on Large Operational Retrieval Systems. In *Res. & Dev. in Inf. Ret., Proc.*, Berlin, May 18-20, 1982, ed. by G. Salton and Hans-Jochen Schneider, Springer-Verlag, Berlin, 38-50, 1983.
- [NAIS 84] Naish, L. *MU-Prolog 3.1db Reference Manual*. Melbourne Univ., 1984.
- [OCON 80] O'Connor, J. Answer-Passage Retrieval by Text Searching. *J. Am. Soc. Inf. Sci.*, 31(4):227-239, 1980.
- [ODDY 77] Oddy, R.N. Information Retrieval Through Man-Machine Dialogue. *J. Doc.*, 33(1):1-14, March 1977.
- [PATE 84] Patel-Schneider, P.F., R.J. Brachman, and H.J. Levesque. ARGON: Knowledge Representation meets Information Retrieval. Fairchild Technical Report No. 654, FLAIR Technical Report No. 29, Sept. 1984.
- [PEEL 85] Peels, Arno J.H.M., Janssen, Norbert J.M., and Wop Nawijn. Document Architecture and Text Formatting. *ACM Trans. Office Inf. Sys.*, 3(4):347-369, Oct. 1985.
- [PERE 83] Pereira, F. Logic for Natural Language Analysis. Tech. Note 275, SRI Int., Jan. 1983.
- [SAGE 75] Sager, N. Sublanguage Grammars in Science Information Processing. *J. Am. Soc. Inf. Sci.*, 26(1):10-16, Jan.-Feb. 1975.
- [SALT 72] Salton, G. A New Comparison Between Conventional Indexing (Medlars) and Text Processing (SMART). *J. Am. Soc. Inf. Sci.*, 23(2):75-84, 1972.
- [SALT 75] Salton, G., Wong, A., and C.S. Yang. A Vector Space Model for Automatic Indexing, *Commun. ACM*, 18(11):613-620, Nov. 1975.
- [SALT 80] Salton, G. The SMART System 1961-1976: Experiments in Dynamic Document Processing. In *Encyclopedia of Library and Information Science*, 1-36, 1980.
- [SALT 83b] Salton, G., Fox, E.A., and Wu. H. Extended Boolean Information Retrieval. *Commun. ACM*, 26(11):1022-1036, Nov. 1983.
- [SALT 85] Salton, G., Fox, E.A. and E. Voorhees. Advanced Feedback Methods in Information Retrieval, *J. Am. Soc. Inf. Sci.*, 36(3):200-210, May 1985.
- [SHAR 86] Sharan, Sharat. Information Retrieval Comparisons. MS Report, Virginia Tech Dept. of Comp. Sci., Blacksburg, VA, June 1986.
- [SPAR 84] Sparck Jones, K. and J.I. Tait. Automatic Search Term Variant Generation. *J. Doc.*, 40(1):50-66, March 1984.
- [SUBR 85] Subrahmanyam, P.A. The "Software Engineering" of Expert Systems: Is Prolog Appropriate? In *IEEE Trans. on Software Engineering*, 11(11): 1391-1400, November 1985.
- [TEN0 84] Tenopir, Carol. Full-Text Databases. *ARIST*, 19:215-246, 1984.
- [THOM 85] Thompson, R.H. and W.B. Croft. An Expert System for Document Retrieval. *Proc. Expert Systems in Gov. Symp.*, IEEE, 448-456, Oct. 1985.
- [TONG 83] Tong, R.M. et al. A Rule-Based Approach to Information Retrieval: Some Results and Comments. *Proc. AAAI-83*, 1983.

- [WEYE 84] Weyer, S.A., and A.H. Borning. A Prototype Electronic Encyclopedia. In *ACM Trans. on Office Information Systems*, 3(1): 63-68, January 1984.
- [WHIT 83] White, C. The Linguistic String Project Dictionary for Automatic Text Analysis. In *Proc. Workshop on Machine Readable Dictionaries*, SRI, Menlo Park, CA, May 1983.
- [WOHL 86] Wohlwend, Robert C. Creation of a Prolog Fact Base from the Collins English Dictionary. MS Report, VPI&SU Computer Science Dept., Blacksburg, VA, March 1986.

AUTHOR AFFILIATION: Edward Fox is an Assistant Professor in the Department of Computer Sciences, Virginia Tech.

ADDRESS: Department of Computer Science
562 McBryde Hall
Virginia Tech
Blacksburg, VA 24060

CODER Data, Users

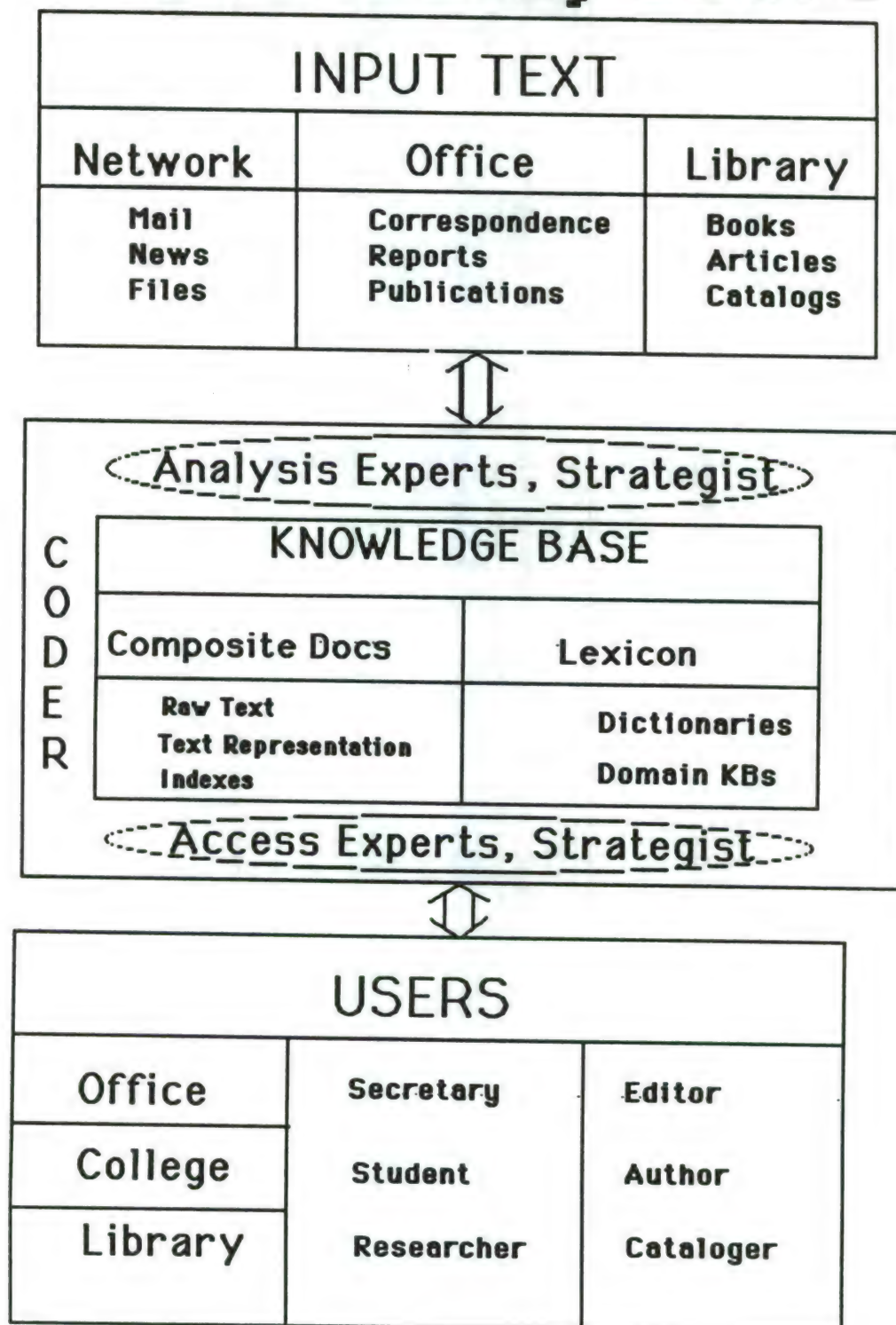


FIGURE 1

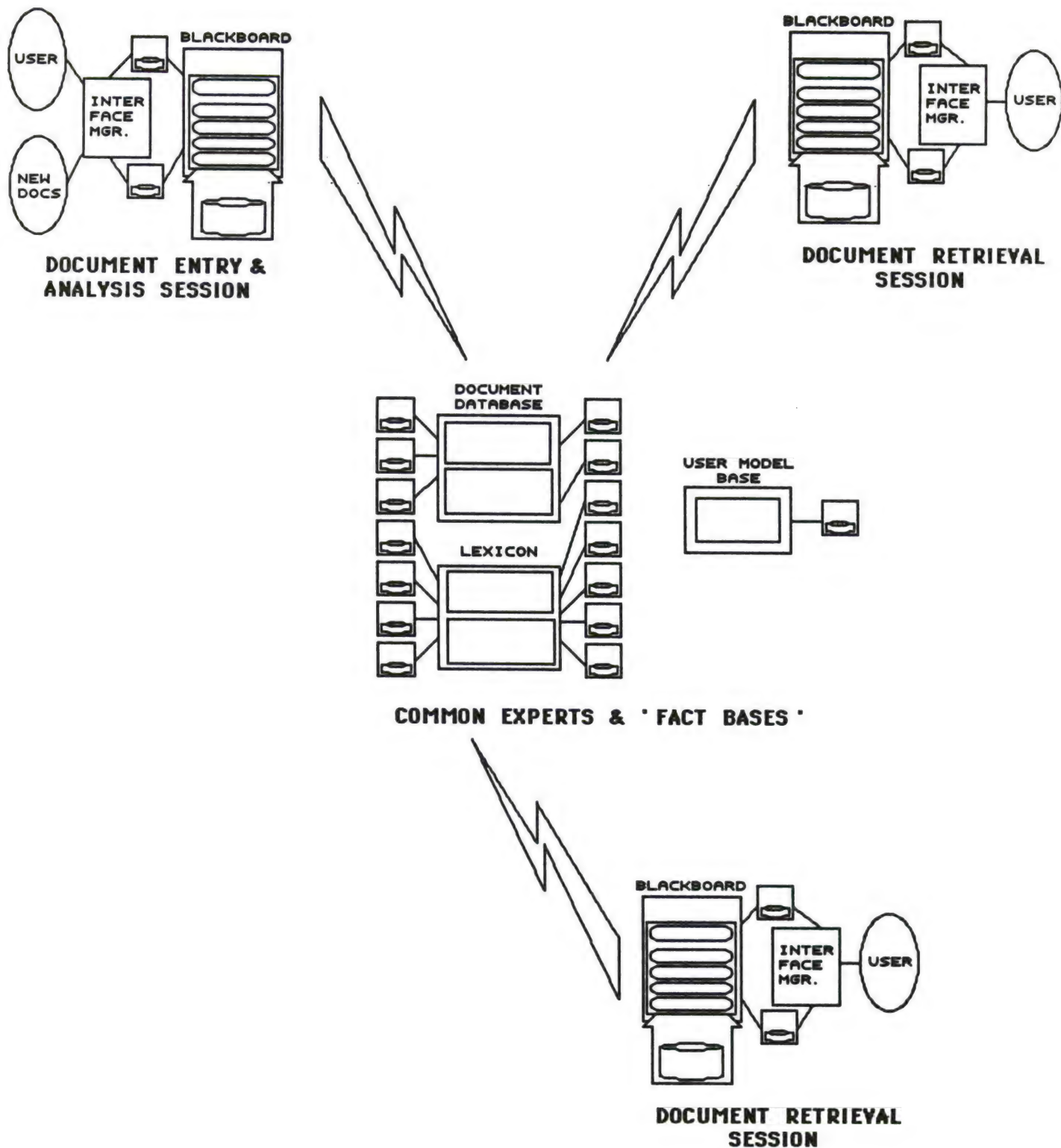
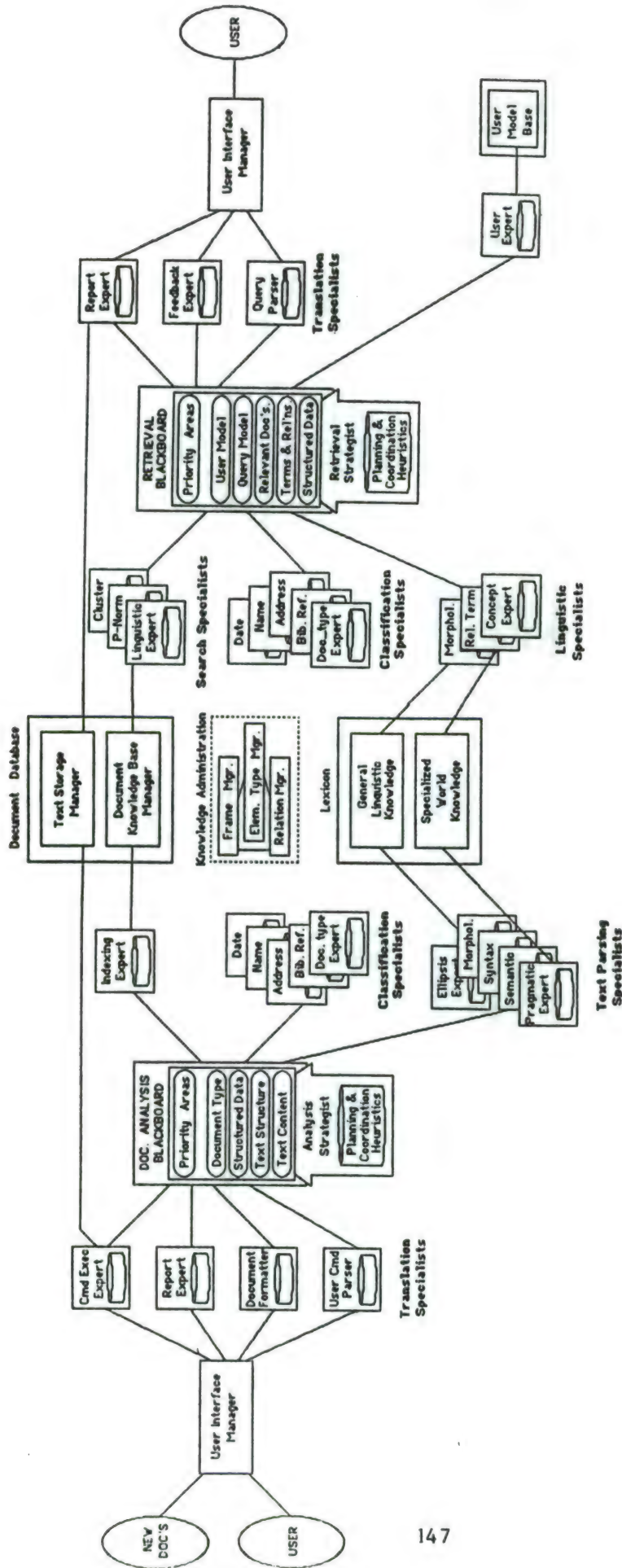


Fig. 2: Overview of system operation in a distributed environment.

ANALYSIS SUBSYSTEM

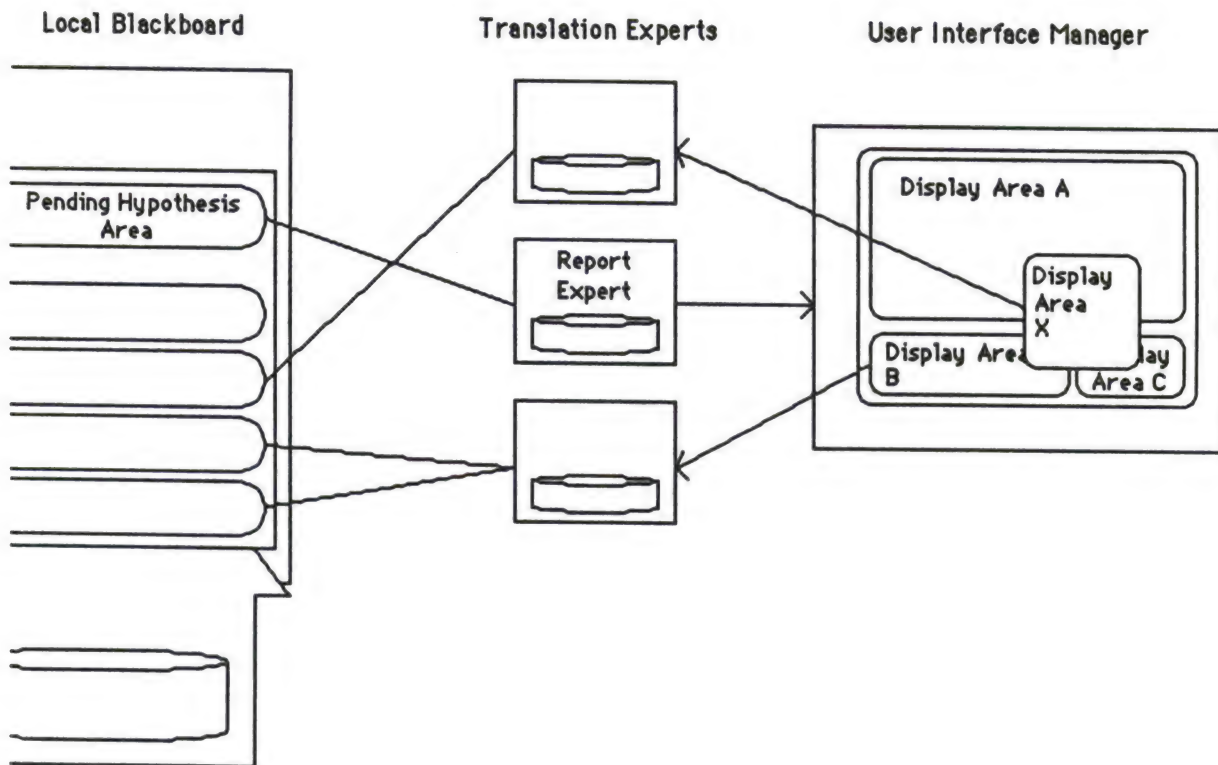
RETRIEVAL SUBSYSTEM



Overview of the CODER System

Figure 3

Figure 4: User Interface Subcommunity



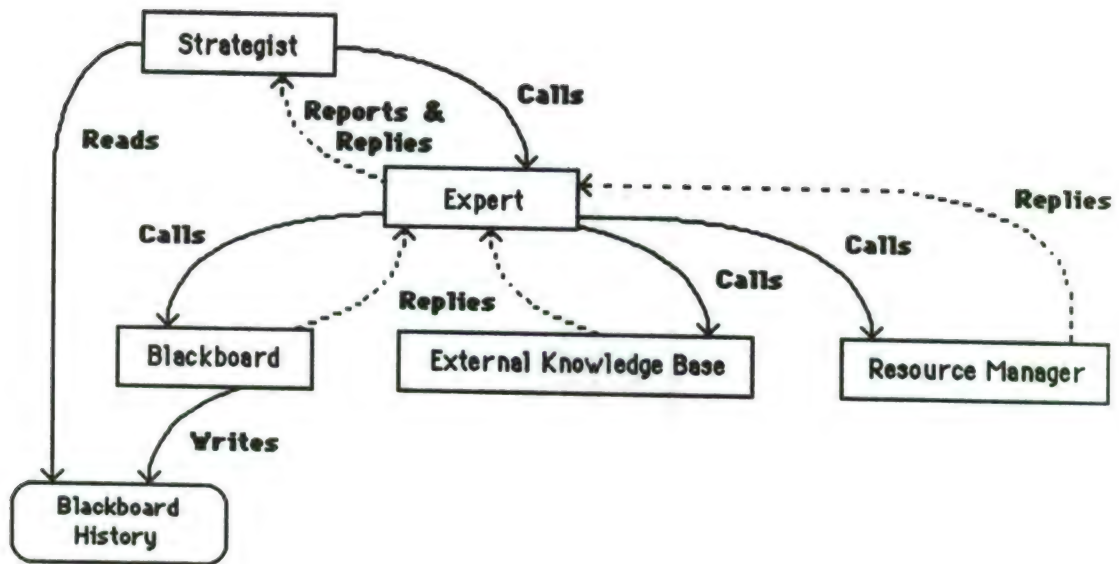


Fig. 5 : A slightly simplified version of the calling hierarchy in a CODER community of experts.

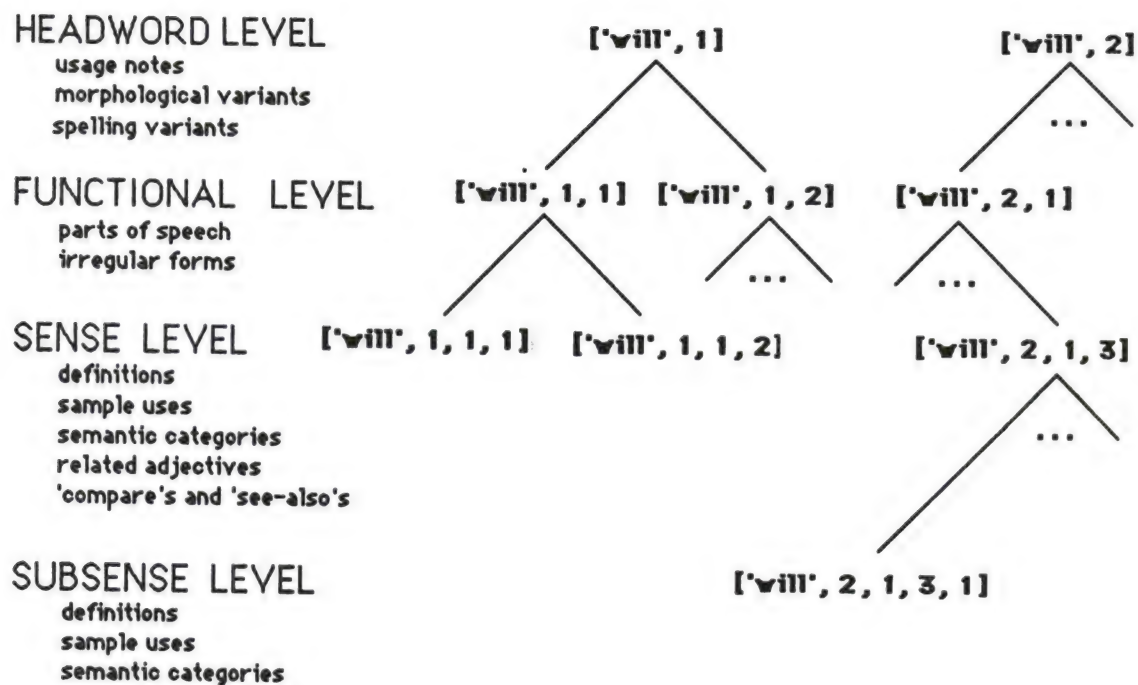


Fig. 6 : Hierarchical structure of terms in the Lexicon, showing knowledge attaching at each level.

The following list presents the syntax of the PROLOG facts:

```
c_ABBREV([ word, homnum, defnum ], relword).
c_ALSO_CALLED([ word, homnum, defnum ], relword).
c_CATEGORY([ word, homnum, defnum ], relword).
c_COMPARE([ word, homnum, defnum ], [ relword, homnum, posnum ]).
c_DEF([ word, homnum, defnum, subnum ], relword, blocknum).
c_DEF_NUM([ word, homnum, defnum, subnum ], relword, totalblocks).
c_HEADWORD([ word, homnum ]).
c_MORPH([ word, homnum ], relword, pos).
c_NLAST([ word, homnum, defnum ], relword).
c_PAST([ word, homnum, defnum ], relword).
c_PLURAL([ word, homnum ], relword).
c_POS([ word, homnum ], pos).
c_RELADJ([ word, homnum, defnum ], [ relword, homnum ]).
c_SAMP([ word, homnum, defnum, subnum ], relword).
c_SINGULAR([ word, homnum, defnum ], relword).
c_SYLL([ word, homnum ], syllword).
c_USAGE([ word, homnum ], relword, blocknum).
c_USAGE_NUM([ word, homnum ], relword, totalblocks).
c_VAR_SPELL([ word, homnum ], relword).
c_VAR_SYLL([ word, homnum ], syllword).
```

where:

word	= 'lexeme' (in single quotes)
homnum	= headword level (integer)
relword	= 'word' or 'phrase' (in single quotes)

Figure 7: Reference Table of Syntax of Relations

Date: 6 October 1985 2023-EDT
From: Hans Berliner
Addr: Berliner@A.CS.CMU.EDU
Subject: Computer chess: hitech
Header: Games - Hitech Chess Performance.

[Forwarded from the CMU bboard by Laws@SRI-AI.]

Hitech won its first tournament, and one with 4 masters in it. It scored 3 1/2 - 1/2 to tie for first in the Gateway Open held at the Pittsburgh Chess Club this week-end. However, on tie break we were awarded first place. En route to this triumph, Hitech beat two masters and tied with a third. It also despatched a lesser player in a brilliancy worthy of any collection of games. One of the games that it won from a master was an absolute beauty of positional and tactical skill. It just outplayed him from a to z. The other two games were nothing to write home about, but it managed to score the necessary points. I believe this is the first time a computer has won a tournament with more than one master in it.

We will have a show and tell early this week.

**A representative message from the
AIList collection.**

When did a chess program last defeat a human opponent?

A representative query.

Figure 8

c_HEADWORD(['win', 1]).
 c_PLURAL(['win', 1], 'won').
 c_POS(['win', 1, 1], vb).
 c_DEF(['win', 1, 1, 1], '(intr.) to achieve first place in a competition', 1).
 c_DEF(['win', 1, 1, 2], '(tr.) to gain or receive (a prize, first place, etc.) in a competition', 1).
 c_DEF(['win', 1, 1, 5], 'to gain victory or triumph in (a battle, argument, etc.)', 1).
 c_POS(['win', 1, 2], n).
 c_CATEGORY(['win', 1, 2, 1], 'Informal').
 c_DEF(['win', 1, 2, 1], 'a success, victory, or triumph'

 c_HEADWORD(['triumph', 1]).
 c_POS(['triumph', 1, 1], n).
 c_DEF(['triumph', 1, 1, 1], 'the feeling of exultation and happiness derived from a victory or major achievement', 1).
 c_DEF(['triumph', 1, 1, 2], 'the act or condition of being victorious; victory', 1).

 c_HEADWORD(['beat', 1]).
 c_PLURAL(['beat', 1], 'beat').
 c_POS(['beat', 1, 1], vb).
 c_DEF(['beat', 1, 1, 11], 'to overcome (an opponent) in a contest, battle, etc.', 1).

 c_HEADWORD(['dispatch', 1]).
 c_VAR_SPELL(['dispatch', 1], 'despatch').
 c_POS(['dispatch', 1, 1], vt).
 c_DEF(['dispatch', 1, 1, 4], 'to murder or execute', 1).

 c_HEADWORD(['out-', 1]).
 c_POS(['out-', 1, 1], prefix).
 c_DEF(['out-', 1, 1, 1], 'excelling or surpassing in a particular action', 1).
 c_SAMP(['out-', 1, 1, 1], 'outlast').
 c_SAMP(['out-', 1, 1, 1], 'outlive').

 c_HEADWORD(['play', 1]).
 c_POS(['play', 1, 1], vb).
 c_DEF(['play', 1, 1, 2], '(tr.) to contend against (an opponent) in a sport or game', 1).
 c_SAMP(['play', 1, 1, 2], 'Ed played Tony at chess and lost.').

Fig. 9: Lexical relations relevant to identifying the conceptual clusters.

Natural Language Querying of a Medical Knowledge Base

Lionel M. Bernstein

Given the massive amounts of new publications and the variable quality of published data, medical practitioners cannot read and process all the information relevant to their patients' problems. To overcome these problems, a Gastrointestinal Disease Knowledge Base has been developed as the first module of a system for all of internal medicine. ASK*MED (Accessible Stores of Knowledge in MEDicine) is an integrated computerized Knowledge Base derived from selected high quality past and recent medical literature that practicing physicians can easily and rapidly access for information to support their diagnostic and therapeutic decisions. It has an updating mechanism; can be easily accessed via a natural language query system; and provides browsing capabilities and contextual understanding in the areas of information provided in response to queries.

The retrieval system combines use of probabalistic, linguistic, and empirical means to rank the relevance to the query of individual text paragraphs and is very effective in presenting the most relevant information first. The knowledge retrieval system will be compared with some cognitive psychological aspects of locating information in full textual databases, and will be related to selected aspects of artificial intelligence and "expert" systems.

Knowledge System, Medicine, Gastrointestinal Disease, Natural Language Processing, Online, Information Retrieval, Artificial Intelligence, Expert System

NATURAL LANGUAGE QUERYING OF A MEDICAL KNOWLEDGE BASE

1. Introduction

Medical decision making can be divided into two broad (although admittedly intermeshed and inseparable) stages. The first deals with the state of knowledge of the physician, embodying both the amount of the physician's knowledge and the internal cerebral "structure" or "representation" of that knowledge; together, this becomes one individual's representation of the state-of-knowledge. The second stage consists of processing data from individual patients to reach diagnostic, therapeutic and prognostic decisions. During the second stage, in idiosyncratic ways, the physician engages in a thinking process which merges the individual's understanding of the state-of-knowledge with the data related to a given patient circumstance to reach a conclusion. Arbitrarily it is accepted as a generality that the best decision making is based on the best understanding of the underlying state-of-knowledge regarding the physiologic and disease processes of an individual patient or problem. The interaction between these two stages is obvious. In the process of contemplating decisions, the need for additional understanding of the underlying state-of-knowlege may become evident; in turn, this often leads the decision maker to return to sources of such information and knowledge, the obtaining of which consequently influences the decisions reached.

The purpose of this presentation is to describe some manners in which the underlying state of knowledge is sought during the decision making processes of practicing physicians, and the potential role of natural language querying of a computerized Knowledge Base.

2. Interaction of the State-of-Knowledge with a Model of the Decision-Making (Thinking) Process

In the medical decision-making process, there is a task environment (TE) which consists of the patient problem (e.g. a diagnostic or therapeutic decision which needs to be made). Imbedded in that task environment are varying amounts of data, ranging from that gained in the initial encounter (the patient's history and physical findings) to a later much more complex array of augmented information including laboratory and X-ray findings, results of procedures (e.g., endoscopy, cardiac catheterization, etc.) and other data. Within this task environment there is a "SCHEMA" which is the perception of the task by the physician. Next there is a stage of ACQUISITION in which the physician surfaces the information/knowledge required for addressing the task; this occurs by drawing upon his or her memory, and/or obtaining it from other sources (e.g., from the institutionalized biomedical literature). There is some perceptual processing of the task via which physicians decide that they need more information than that they have in their own memory, or that they can proceed with the task relying only on their own internal knowledge. It is at this crucial acquisition stage that the utility and ease of use of information sources become critical for the physician who identifies the need for more information than he has from his own cognitive memory.

The INFORMATION PROCESSING stage then proceeds. In the course of that information processing many factors are brought to bear which influence the quality and effectiveness of the ultimate decisions reached. Such factors (with obviously much overlap) include:

1. Individual personal capabilities: there is great variability among physicians as to their general intelligence, nature and duration of training, attitudes, philosophy, etc.,
2. Knowledge structure of the brain: physicians have greatly varying amounts of knowledge about the field of medicine.
3. Utilization of clinical experience: the unique perspective and understanding of an individual physician may be brought to bear and result in conclusions not reached by anyone else for a given problem.
4. Framework of the individual physician capable of accepting "plug-ins" of knowledge.
5. The influences of flawed cognitive behavior (bias, limits of memory and cognitive power, information overload, undue emphasis on salient cases, framing, etc.)

The consequence of the described information processing stage is the formulation of an OUTPUT and ACTION: the decisions (the actions) result in outcomes which may through feedback pathways subsequently modify both the actual task environment as well as the physicians schema for understanding the problem. Indeed, iterations progressively alter the nature of the problem as it is dealt with in a series of subdecisions which constitute the overall major decision.

3. The Acquisition of Information Needed in the Decision-Making Process

Assume a physician practitioner faced with a gastrointestinal disease patient problem recognizes the need for an external more current view of the state-of-knowledge regarding that problem, and seeks such information. Table 1 provides four examples of sources to be searched for that information, each of which can be thought of as an "information task environment" in which the task problem is to obtain the general state-of-knowledge that is applicable to the patient problem. The physician has a schema or perception as to the ability of each of those sources of information to provide the needed knowledge, and

Table 1. Acceptability of Various Medical Information Sources for Daily Onsite Use by Clinical Practitioners

Attributes of Information Sources*	Information Sources					
	Health Sciences Library	100 Books Plus 300 Journals	2000 Page Multi-authored Subspecialty Textbook	200 Page Subspecialty Treatise	Biblio-graphic Searches	ASK*MED
	Scale: Very acceptable = 10; Very unacceptable = 1					
Comprehensiveness of Coverage	10	9	6	3	9	7
Depth/Detail of Coverage	10	9	6	1	7	7
Availability of Recent Material	10	9	4	4	10	7
Redundancy	1	2	9	10	2	8
Access to Information						
Effort-time required	1	2	9	10	1	9
search efficiency	1	2	9	10	1	9
Authoritativeness	3-10	5-10	10	9	3-10	10
Weinburg Criteria	1	2	7	3	1	8

*See text for more detailed explanation of attributes.

decides amongst them. In Table 1, the four sources of information represent 1) a 200 page summary treatise on gastrointestinal disease (e.g., the section of a leading textbook of internal medicine); 2) a 2000 page leading subspecialty textbook of medicine (e.g., on gastrointestinal disease); C) 100 books plus 300 journals (as in a departmental library); and D) an entire health sciences library with all of its sources and media, including online capabilities for access to distant data bases. Although these four options are only samples of the many more available, they serve the purpose of demonstrating how the practitioner needing medical information pursues obtaining that information once the decision to do so has been made.

A. Attributes of Four Medical Information Sources

Table 1 summarizes some attributes of the four information sources listed above which play important roles in the medical practitioner's decision of which sources to use, or, contrary-wise, in the decision to finnese seeking the needed information. The individual attributes are rated from the perspective of a medical practitioner's daily onsite use of each of the sources to answer the many questions that arise daily in the course of seeing patients. Each of the four sources is graded on a scale from 10 (very acceptable) down to 1 (very unacceptable). Consider the entire health sciences library as the base for comparision; clearly the breadth of coverage of a field (e.g., how extensively and completely gastroenterology diseases are covered), the depth of detail of that coverage, and the presence of recent updating information are maximal. On the other hand, the library also provides the ultimate in undesirable redundancy. Because of its often distant location and the mass of material available to search, it requires levels of effort and amounts of time to meet their many daily information needs that

generally are unacceptable to the practitioners; overall, library searches are seen by them as inefficient. The next largest information source choice in Table 1 (100 books plus 300 selected journals) essentially is a small library, and acceptability of its attributes is judged to be similar to those of the entire health science library, but to slightly lesser extremes.

Single small sources (such as the 200 page treatise or the 2000 page multi-authored subspecialty text) necessarily provide less breadth of coverage of any given field, as well as less depth. However, from a clinical practitioner's view, the 2000 page multi-authored subspecial text has a very respectable level of breadth and depth, and is very advantageous when compared with libraries because of the minimal level of information redundancy. The lesser redundancy of single texts contributes greatly to the great ease and efficiency of access. By deliberate selection, a single very high quality superb text can be used, and can provide the highly authoritative and minimally controversial syntheses of the state-of-knowledge that practitioners need. Obsolescence of the textual content (that is, the lack of updating) is the major objection to single hard copy texts; small percentages of content cumulatively become obsolete each year from the time of publication.

An overview measure of the usefulness of an information source is the extent to which it meets the fundamental task of the information chain as stated by the 1963 Weinberg Report of the Provident's Science Advisory Committee, "...as the switching of information, not documents. The ultimate aim is to connect the user, quickly and efficiently, to the proper information and to only the proper information...", "...the switching system must always allow for some browsing in neighboring areas ...;" "... to be effective, the system must select, compact, and review material for the individual user so

that he actually assimilates what he is exposed to and is not exposed to too much that is unimportant or irrelevant ...". Of the above options presented in Table 1, this goal is most nearly met by the 2000 page sub-specialty text.

Perusal of Table 1 indicates that for the active busy practitioner of medicine each of the medical information sources has limitations in meeting his or her daily worksite information needs. However, when considering an overall composite judgement, the 2000 page subspecialty text stands out as an excellent information source. It is the only one of the four options which has no rankings at the "very unacceptable" end of the scale. It offers a practical compromise between the desirable attributes of a library (comprehensiveness, detail, updating) and the library's necessarily consequent undesirable aspects (redundancy, difficulty of access). The sub-specialty text is the single source that most satisfies the Weinberg criteria described above.

B. Methods for Searching Medical Information/Knowledge Sources

For all four options presented in Table 1, the major mode of searching is manual. It is recognized that online search techniques are commonly used in libraries and via individually owned terminals, but the bulk of searching remains manual.

Searches vary with the nature or characteristics of the sources used; with the complexity of the question asked; and with the level of expertness of the searcher. Searching techniques vary according to the extent of expertness in the content domain, the level of expertness as a searcher, per se (e.g. of a librarian intermediary searcher), and combined expertness in both content domain and search techniques. However, it is of crucial importance to remember that the vast proportion of needed (not synonymous with actual)

searches are by the large part of the scientific population which consists of those seeking information about which they are not expert.

1. Searching in a Single Text

Consider the mechanisms available for searching for information in a given single text, using the 2000 page text on Gastrointestinal Disease by Sleisenger-Fordtran as an example; assuming that the domain of the inquiry was gastrointestinal disease, the selection of this text is appropriate.

Searching occurs via the Table of Contents, the Index, and by browsing.

a. Searching Via the Table of Contents

The Table of Contents provides a structured representation of the main themes of a text. Usually it is hierarchically organized, and often has redundancy because the same content is described from different perspectives. The logic of the hierarchical arrangement of content is a by-product of the care with which the author has organized the material (in turn reflecting the internalized knowledge structure within his brain and the organization of that knowledge) as well as the extent to which the extant knowledge provides for a logical development of the domain. The Table of Contents is scanned for presence in the headings of selected important keywords of the query. Implicit is a ranking of the importance of those individual keywords in locating headings that may be relevant. To the extent that the searcher is cognizant of the general structure of the domain, and that structure is utilized in the arrangement of content within the text, there will be a matching of these similarities making it easier to locate headings that are relevant and to bypass those which are not. Identification of potentially useful headings from the Table of Contents relies primarily on individual keywords of the query, and, as headings are selected as potentially relevant,

they are intellectually evaluated using the syntactical and semantic aspects of the full heading. In the process, the human equates cognates and synonyms of the keywords in judging relevance to the query, and thus performs somewhat automatically both the truncation function and the thesaurus function of computerized search mechanisms. When considered relevant, the user obtains from the Table of Contents heading a page number for detailed intellectual analysis of the full text under it.

b. Searching Via the Index

Use of the Index is an alternative to the Table of Contents for identifying the location of paragraphs whose content is responsive to the query. It consists of an alphabetical arrangement of major and minor concepts as well as some miniscule details contained within the text. It is considerably more extensive and detailed than is a Table of Contents. The index terms are arranged alphabetically for ease of location; often include varying sub-levels of hierarchical structure (within which individual items are presented alphabetically); and present the contents with considerable redundancy. Redundancy is an important positive attribute of an index. The searcher seeks in the Index the occurrence of query keywords, usually assigning importance (weighting or ranking) of those words by sequence of searching (that is, looking at the prime interest words first and the secondarily important words as subsets). It is a process which in the main is void of semantic or syntactical analysis. The human easily identifies and uses cognates (the analog of the truncation mechanism of a computer algorithm); often will identify synonyms and use them as guides to further index searching; and, because of his knowledge of synonyms, be reminded to use other nodes for accessing the text.

In contrast with the Table of Contents, the Index provides a poor representation of the overall domain structure. As with the Table of Contents, the location of relevant Index terms provides a number of a page to which to turn and intellectually explore in detail it's paragraphs.

c. Browsing-for-Finding

The preliminary stages of searching via the Table of Contents or the Index leads the user to specific numbered pages in the text containing paragraphs in which to search in more detail for the desired information. Having been pointed to a given page or pages, local browsing is used for both rapid location of potentially relevant information, and the sorting of those paragraphs (or parts of paragraphs) for which the effort of detailed intellectual analysis should be is expended. The user scans the page for the key words, first in the headings, and then in the body of the text, the keywords appearing in headings being considered more important. If gross scanning of the page reveals headings, those whole headings are intellectually analyzed for relevance to the query; and, if relevant, more indepth scanning of the subtended paragraphs follows. In the absence of headings or the absence of relevant headings, the paragraph bodies themselves are scanned for the key text words (or their cognates or synonyms). The authors and editors have contributed to the ease of this browsing-for-finding process by selectively using italics, bold-type, or underlining for important words or phrases. Such hints of importance lead the searcher to explore the whole sentence in which those emphases have been placed. Useful measures of a paragraph's potential relevance are the number of times a key word is repeated and the presence of increasing numbers of the different search keywords. Another search technique, is to explore the first sentences of paragraphs which frequently serve as reduced cues to the content of the entire paragraph.

Browsing-for-finding is a process which deals serially with one paragraph at a time, and leads to full intellectual analysis of the paragraph or its parts for the information needed. Having intellectually analyzed the paragraph, the user concludes that the information is acceptable, or that the information is inadequate (which in turn may lead either to continued searching or to abandonment of the search).

d. Browsing-for-Context

Separate from browsing-for-finding is "browsing-for-context." Once having obtained information that is relevant to the query, the searcher may choose to gain greater understanding of surrounding areas of information, or seek a broader contextual understanding of the realm in which the query answer was located by browsing to adjacent and/or distant textual areas. To do so, the user moves about the text, guided by the hierarchical structure in either the Table of Contents or local headings.

2. Searching in Multiple Texts

Manual searching of multiple texts requires repeated use of the same process described for searching a single text. Whatever the limitations of the modes for accessing the information (Table of Contents, Index, Browsing-for-finding, Browsing-for-context), the difficulties become exponentially greater as the Knowledge Base consists of progressively larger amounts of information located in multiple texts. In such circumstances, all of the difficulties of locating information are endured multiple times as manual searching is repeated for each separate text. The resulting search process is progressively more tedious and more inefficient. Use of multiple texts further compounds the user's problem by providing great amounts of redundancy, and, because of divergent points of view, augments the need for

further intellectual synthesis by the user of conflicting points of view; this can be an intellectually rewarding experience to some, and an impractical burden and barrier to use by the busy medical practitioner seeking rapid onsite access to information needed to solve problems.

3. Searching a Computerized Knowledge Base: Use of ASK*MED in Searching for Gastrointestinal Disease Information

a) Brief Summary of ASK*MED[™] and ASK*WARE*[™]

The principles and development of ASK*MED (Accessible Store of Knowledge in Medicine) has been previously described and demonstrated to this assembly. For the medical practitioner ASK*MED provides a support mechanism for obtaining a high quality representation of the state-of-knowledge of gastrointestinal disease regarding the patient problems with which he is concerned. ASK*MED is derived from the published biomedical literature. (Dr. Louis Y. Korman contributed to its development). The source material from which ASK*MED is derived is:

- Sleisenger MH, Fordtran JS, eds. Gastrointestinal Disease: Pathophysiology, Diagnosis, Management, 3rd ed. Philadelphia: W.B. Saunders Co.; 1983.
- Greenberger NJ, Moody FG. The 1984 Year Book of Digestive Diseases. Chicago: Year Book Medical Publishers, Inc.; 1984.
- Greenberger NJ, Moody FG. The 1985 Year Book of Digestive Diseases. Chicago: Year Book Medical Publishers, Inc.; 1985.

ASK*MED provides a compacted, constrained, high quality, minimally redundant, authoritative, and updated representation of the state-of-knowledge about gastrointestinal disease. Further, it provides requested information in

*ASK*MED and ASK*WARE are proprietary products of Knowledge Systems Inc., McLean, Virginia.

synoptic form, and in multiple levels of detail so as to be better able to match the highly variable information needs of users (i.e., ranging from medical students to generalists and subspecialists). ASK*MED also allows browsing (going up, down, or sideways in the hierarchically arranged content), thereby providing contextual understanding. These characteristics stem from the deliberate design of the Knowledge Base to address and provide some solution to all of the categories of impediments to diffusion of scientific information that are well characterized in the information science literature.

ASK*WARE (developed by Dr. Robert E. Williamson), the software used in ASK*MED, is a complex but flexible algorithm which provides very easy to use access to the information in the Knowledge Base in response to typing an inquiry on the keyboard in common everyday "natural" language English. The system selectively searches the Knowledge Base, and ranks the individual items (or paragraphs) for their similarity and responsiveness to the query. It then presents fully ranked results, enabling the user to see that text "most likely to be relevant" before "less likely" material. Items containing significant answers to typical questions routinely occur in the top three ranked items.

The essence of the algorithm is based on the treatment of each paragraph (or item) in the Knowledge Base as having a vector in multi-dimensional space, one dimension representing each contained unique word root (excluding common-words deleted via a "stop-word" list). An entered user's query similarly is treated as having a vector in multi-dimensional space, and the search process seeks, locates and ranks those paragraphs whose vectors are most congruent with the vector of the query. Atop this essence, the algorithm contains many empirical, heuristic, and "witchcraft" modifications and

modulations to improve its performance. It is in these modifications of the algorithm that many examples of mimicking human search behavior can be recognized.

b) Minicking Human Search Behavior Combined with Superhuman Computer Capabilities

After a natural language query is entered on the terminal keyboard, the ASK*MED search algorithm performs some functions that mimic human search behavior and does others that offer the special superhuman capabilities of a computer. In the search process, ASK*MED automatically uses cognates and synonyms of the key words in the query. The system mimics the human text searching behavior by increasing the importance (by weighting) of the query key words if they appear in headings (or in first sentences in paragraphs which have no headings) over the weights assigned if they appear in the body of a paragraph. The system also weights for the frequency of occurrence of each individual key word in an individual paragraph (thereby providing an additional dimension beyond that offered by a Boolean search which is limited to simply identifying the presence or absence of key words). Because the system matches for similarity of paragraphs to the query, and finds the most similar answers, it does not require that all key words of the query be present (adding another desirable dimension that Boolean systems do not have). When the search is completed the system presents the most relevant paragraphs first. This, too, is a great advantage over Boolean systems in which the "hits" are presented to the user in random order (usually in reverse chronological order of the dates the items were entered into the file being searched), and the burden is then placed on the user to scan the output to locate desired items (which may be at the beginning, at the end, or anywhere inbetween).

ASK*MED facilitates intellectual analysis of individual paragraph content by providing inverse video and bold highlighting, respectively, of the actual key words of the query and of their cognates. The user's attention is thereby quickly drawn to that portion of displayed headings or paragraphs most likely to be relevant, and reduces the amount of time and effort expended on scanning the portions of paragraphs not responsive to the query. This is particularly important for perusing lengthy paragraphs.

The inverted file of the text used in ASK*MED (consisting of all non-common words; that is, those not on the stop-word list) is much more extensive than would be a manually constructed index. It therefore provides access to much more detailed information than would either a Table of Contents or an Index.

c) Hierarchical Browsing and Contextual Understanding

A major advantage of ASK*MED and ASK*WARE is the provision of contextual understanding by the enhancement of the underlying hard copy text. The hierarchical structure of multiple and varying levels of detail of the content allows browsing to the level of detail appropriate to the individual searcher. Further, the ability to display the local structure of the domain (a local Table of Contents surrounding the answers displayed in a response to a query) provides an opportunity for seeking contextual understanding beyond that which is possible from the underlying texts from which the Knowledge Base was derived.

d) Stored Searches: Less Harried Review and Study of Medical Information

Another important attribute of ASK*MED is the ability to store searches for later perusal. Although the system is indeed primarily designed to support onsite decision-making by physicians when encountering patients by

providing the underlying state-of-knowledge requested with negligible time delays, it also supports less hurried educational opportunities. The use of a printer also allows any part of the displayed information considered relevant to be retained in hard copy for later reading and perusal, and for insertion in medical records to document the search for the basic state-of-knowledge relevant to individual patient problems.

e) ASK*MED as a Selective Bibliographic Tool

It is axiomatic that for any given patient problem the physician should become as close to fully aware of all the information relevant to that problem that the problem demands. It is believed the amounts of information within ASK*MED will answer a very large majority of practitioner's questions, particularly as the practitioner moves down the information hierarchy to the greatest detail available. However, it is also evident that for some questions the practitioner must access kinds and details of published knowledge that are beyond the scope of ASK*MED. ASK*MED is geared to facilitating and encouraging the individual physician to seek selected information beyond its content by providing pointers to that primary literature that was used by the authors to document their "syntheses" of the state-of-knowledge. Because these citations are linked to the lowermost level paragraphs of the hierarchical structure in ASK*MED, on request (by typing "Ref") the references cited in the text paragraphs are displayed in MEDLINE format, and serve as efficient guides to the practitioner to enter selectively and efficiently the highest quality of the mass of primary literature in the library.

f) Updating

The major defect of subspecialty textbooks is their obsolescence. This is significantly remedied by ASK*MED by inclusion of distillates from the more recent literatures from the YEAR BOOKS of DIGESTIVE DISEASE to augment the Sleisenger-Fordtran text. The YEAR BOOKS information covers the recent two years of published information to complement the Sleisenger-Fordtran text coverage of the preceding decades of information. As seen for the ratings given ASK*MED in Table 1, ASK*MED improves the "updating" acceptability beyond that of the 2000 page subsequently textbook.

4. Conclusion

In theory, and in abstract contemplation, a system such as ASK*MED appears to have all of the attributes required to meet the information needs of busy medical practitioners. Simple natural language querying with timely responses is an essential attribute for accessing the high quality information needed in their daily decisions. This subjective optimism must be displaced by objective evaluation to ASK*MED; field trials are about to get underway in several clinical settings. The usefulness of a computerized Knowledge Base such as ASK*MED will be defined by the judgments of its utility by busy practicing physicians who use it. If results of the field tests are successful, they will form the basis for extending efforts to augment the domain of ASK*MED to include the other subspecialties of internal medicine.

AUTHOR AFFILIATION: Lionel Bernstein is Professor, Health Professions Education, Center for Educational Development, University of Illinois at Chicago.

ADDRESS: University of Illinois at Chicago
808 S. Wood Street
Chicago, IL 60612

THE DoD GATEWAY INFORMATION SYSTEM: PROTOTYPE EXPERIENCE

Gladys A. Cotter

The Department of Defense (DoD) Research and Engineering community requires rapid, easy access to scientific and technical information relevant to its mission. This information is contained in a multiplicity of databases maintained within the federal and commercial sectors. The DoD Gateway Information System (DGIS) is being developed to provide this community with a modern tool for accessing these databases and extracting information products from them.

DGIS is designed to provide the DoD researcher with a single, user-friendly system front-end that can be used to identify, access, interrogate, and post-process information from numerous databases. The DGIS transforms these database and processing resources into a single entity -- a virtual database.

The Defense Technical Information Center (DTIC) is responsible for managing the design, development, and implementation of DGIS. A prototype version of DGIS is now undergoing test and evaluation. Eight information services have been selected as targets of the prototype gateway.

The utility of DGIS will rest on the early acceptance and widespread participation of users: reactions encouraged by a design that makes quick learning and valuable results both possible and obvious. DGIS system development guidelines emphasize these design characteristics. The development effort is concentrated on integrating within a single, straightforward command structure the differing logic, syntax, and procedures intrinsic to separate databases. The design accommodates, transparent to the user, the complexities of accessing, downloading, merging, and processing information from diverse sources. This paper describes DGIS developments and evaluations to date.

Intelligent Gateways, Front-End, User Interfaces, Networking

THE DoD GATEWAY INFORMATION SYSTEM:
PROTOTYPE EXPERIENCE

by
Gladys A. Cotter

Keywords: Intelligent Gateways, Front End, User Interfaces, Networking

Abstract: The Department of Defense (DoD) Research and Engineering community requires rapid, easy access to scientific and technical information relevant to its mission. This information is contained in a multiplicity of databases maintained within the federal and commercial sectors. The DoD Gateway Information System (DGIS) is being developed to provide this community with a modern tool for accessing these databases and extracting information products from them.

DGIS is designed to provide the DoD researcher with a single, user-friendly system front-end that can be used to identify, access, interrogate, and post-process information from numerous databases. The DGIS transforms these database and processing resources into a single entity - a virtual database.

The Defense Technical Information Center (DTIC) is responsible for managing the design, development, and implementation of DGIS. A prototype version of DGIS is now undergoing test and evaluation. Eight information services have been selected as targets of the prototype gateway.

The utility of DGIS will rest on the early acceptance and widespread participation of users: reactions encouraged by a design that makes quick learning and valuable results both possible and obvious. DGIS system development guidelines emphasize these design characteristics. The development effort is concentrated on integrating within a single, straightforward command structure the differing logic, syntax, and procedures intrinsic to separate databases. The design accommodates, transparent to the user, the complexities of accessing, downloading, merging, and processing information from diverse sources. This paper describes DGIS developments and evaluations to date.

1. **INTRODUCTION**

The Defense Technical Information Center (DTIC), for many years, has been charged with collecting from, and disseminating to, the Department of Defense (DoD) research information and reports generated by DoD and its contractors. This has been accomplished, in part, through the development and maintenance of the Defense Research, Development, Test and Evaluation Online System (DROLS). Citations of technical reports produced by the DoD, up to the "Secret" level, are contained in DROLS and may be retrieved by remote users via terminal. Unclassified DROLS information is provided to the National Technical Information Service (NTIS) and made available to the public.

The DTIC mission recently was expanded to include facilitating DoD access to external Scientific and Technical Information (STI) databases, online services, and networks relevant to DoD Research and Engineering Programs. The STI sources are to include other federal, commercial, and foreign databases and systems. The objective is to provide DoD with the most up-to-date, pertinent information available, regardless of source.

To meet this additional mission, DTIC elected to develop the DoD Gateway Information System (DGIS), an intelligent gateway. This intelligent gateway is to provide a simple mechanism for matching information needs with information. The gateway is to provide users with answers to the questions:

WHAT RELEVANT DATABASES EXIST?

HOW DO I ACCESS THEM?

HOW DO I RETRIEVE INFORMATION?

HOW DO I MANIPULATE THE RESULTING INFORMATION?

Stated another way, DGIS is to provide a single, easy-to-use interface for identifying, accessing, interrogating, and post-processing information from numerous databases relevant to DoD information needs. Development of DGIS is a multi-year, multi-task project. This paper describes development of a prototype DGIS which currently is undergoing test and evaluation.

2. PROTOTYPE INGREDIENTS

Because the project's objective was intimidating, the initial fear was not failure, but never getting started. Failure, after all, is a tangible result which can be improved upon. The best approach seemed to be to identify several achievable objectives, to devise a plan to develop and implement a prototype system encompassing them, and to subject the results to test and evaluation. The basic components of the system were to be:

A DIRECTORY OF DATABASES - SUBJECT SEARCHABLE

A COMMON METHOD FOR ACCESSING AND SEARCHING DIVERSE DATABASES

TOOLS FOR DOWNLOADING AND POST-PROCESSING DATA

These components were to be encased in a simple, eloquent system. DGIS was to be designed for a DoD user community including both intermediaries and end users. Databases would be both federal and commercial. In addition to large, well-known databases and systems, many small, specialized DoD databases would eventually be part of the DGIS. Resources would be required to identify these databases within DoD and to establish announcement and access permission.

Having identified the broad requirements of DGIS, a software survey was conducted to determine if a software product already existed which would meet its needs. The survey showed that "THE" system was not out there, waiting (Ref. 1). One software package, the Technology Information System (TIS), did provide a suitable foundation on which to begin constructing DGIS (Ref. 2). TIS was under development at Lawrence Livermore National Laboratory (LLNL) under the sponsorship of the Department of Energy (DOE) (Ref. 3). TIS functioned as an

intelligent gateway, capable of interconnecting heterogeneous information resources at geographically distributed locations in an automated, unified, and controlled manner. In addition, TIS downloading and post-processing capabilities were already available for selected databases.

Having identified in TIS a hospitable environment for testing DGIS concepts, a strategy for DGIS prototype design and development was based upon its capabilities.

3. PROTOTYPE DESIGN

Project teams were assigned to each of the DGIS major components: Directory of Databases, Interface(s) for Searching Diverse Databases, and Post-Processing Capabilities. Over the course of the project, teams were added for DGIS interface development, user support, and prototype implementation. Each of these areas is described below.

3.1 Directory of Databases

The goal of this project team is two-fold: 1) identify and catalog existing databases and 2) make this information subject-searchable, so that information needs can be matched to relevant resources (Ref. 4). Although there are many directories that identify commercial and prominent federal databases, such information was not readily available for DoD databases. To fill this void, a questionnaire was addressed to the DoD Research and Development (R&D) community to identify extant databases, their scope, and availability. Over 400 databases were identified as a result (Ref. 5).

The next step was to build a database of DoD and DoD-relevant databases which had been identified. A user survey was conducted to determine database requirements (Ref. 6). A database schema was developed for the Directory and database entries were subject indexed. The database was built using the INGRES relational database management system.

The result of this effort is a small-scale, online Directory of Databases which contains information on the content, scope, and availability of selected databases. The Directory is subject-searchable, so that upon entering the topic of interest, the user is provided with a list of appropriate databases.

3.2 Interface for Searching Diverse Databases

One of the primary goals of DGIS is to relieve the user of the need to learn and master separate commands and protocols for each database accessed. As mentioned earlier, the DGIS target user community includes both end users and intermediaries. It is a rare intermediary who maintains proficiency in the use of more than ten systems and, for end users, two systems is high (Ref. 7). With the ongoing proliferation of databases, it is obvious that both end users and intermediaries will benefit from an interface for searching diverse databases. The project team assigned to this effort found that end user and intermediary interface needs are very different when considered in conjunction with today's technology. (An expansive natural language interface requiring artificial intelligence applications appealed to both populations, but could not be accomplished with existing technology in the short term.)

A dual approach was adopted for the interface design, incorporating separate strategies for intermediaries and end users. Eight database systems were selected for inclusion in the prototype model. These included three large federal systems and two small DoD systems.

A software survey was conducted to determine if interfaces existed which met DGIS needs. One of the "needs" that influenced the result of this survey was the requirement to support low-end, "dumb" terminal service on DGIS, as well as intelligent devices. Many of the software packages identified were designed for a microcomputer environment and had to be ruled out for use in the prototype. As a result of the survey, a decision was made to develop an interface for the intermediary and to integrate an existing interface for the end user.

For the intermediary, a command translator is being developed which allows the user to interact with any of the test systems using the command language he selects. The user, for example, could search NASA/RECON using DROLS commands or the reverse. Since some commands will not have an equivalent in another database, native command searching will be retained. This development work is being performed at LLNL under the sponsorship of DoD and NASA.

To satisfy the end user, the EasyNet database searching service has been integrated into DGIS. EasyNet is a menu-driven, database front-end which provides access to over 600 commercial databases. This service was tested by members of the DoD end user community who delighted in the simplicity of search execution. EasyNet access is now an option within DGIS.

3.3 Post-processing

Information retrieved from databases often requires analysis or post-processing in order to become useful to the researcher. This need had been recognized by DOE and they had developed many options for post-processing data from DOE/RECON through TIS. A library of post-processing routines for numeric and bibliographic data was available on TIS software and was incorporated in the DGIS project (Ref. 8). In order to post-process data, the user downloads it into a file on DGIS, translates the data into a common format, and calls up one of the many available post-processing routines.

The post-processing team tested existing post-processing capabilities, made recommendations for enhancements, and prioritized expansion of the capabilities to other databases.

3.4 DGIS User Interface Design

The interface incorporated in TIS software was structured to support users with a knowledge of the UNIX operating system. As a rule, the DGIS user community lacked this knowledge and was not inclined to invest the time required to acquire UNIX expertise. However, TIS software has a flexible design which allows the user interface to be tailored to the target user community (Ref. 9).

The team responsible for developing the DGIS interface incorporated menu and command modes into the interface (Ref. 10). The objective was to allow the novice user to interact with the system at ease, by descending through a series of menus. For the more experienced user, commands were incorporated to execute systems functions.

3.5 Prototype Implementation

Initial DGIS development, test and evaluation took place on TIS software at LLNL. DTIC sponsored a number of DoD users who agreed to test DGIS capabilities being developed at LLNL and make recommendations for system enhancements. These users were issued passwords and dialed into TIS at LLNL.

In parallel, the prototype implementation team developed a plan for acquiring the hardware, software, and telecommunications equipment required to support a DGIS prototype in the Washington, DC, area. A site was selected and installation was accomplished in February 1986. The prototype is running on a VAX 11/780 using the UNIX operating system and INGRES database management system.

3.6 User Support

As implementation and testing of the various DGIS modules began, it became obvious that success of the system would require some form of user support and training. A Gateway User Support and Training Office (GUSTO) was established to satisfy this need. GUSTO provides a hotline service which users can call when they have a problem. GUSTO staff will identify the source of the problem (i.e., the gateway, the user's terminal, a telecommunications link, a remote system, etc.) and take action to have the problem resolved. Users may also contact GUSTO staff utilizing an electronic mail capability which is available on DGIS.

Developing a user's manual and providing training courses are also GUSTO responsibilities. The training course is primarily designed for the professional searcher who wants to exercise the power of the system, especially in the area of post-processing of bibliographic data. The user's manual serves as a reference tool for the user. GUSTO staff will also poll the DGIS user community to identify new requirements.

4. STATUS

The prototype DGIS is currently undergoing test and evaluation within the DoD community. Thirty users have been selected to participate in the prototype evaluation. The group is comprised of end users and intermediaries.

Testing of DGIS began in March 1986 and will continue for a 12-month period. Currently, in the first month of prototyping, the major effort has been devoted to correcting a series of minor, but continuous, hardware and software problems. This has provided an opportunity to gain a first-hand knowledge of the "burn-in period" and "infant mortality."

Once these problems have been solved, the users will be brought onboard and the design teams will begin collecting data on the performance of their modules. System modification and fine tuning will take place to enhance performance. Six months into the prototype period, data will be collected on overall system ease of use and productivity improvements.

Success of the prototype will result in its operational implementation and expansion of the number and type of databases accessible through DGIS. Implementation of DGIS will provide DoD's scientific and technical community with a powerful, responsive information tool. DGIS will render timely, comprehensive information to DoD research, development, and engineering programs. The productivity enhancement within the community resulting from this information will more than offset the investment made in DGIS development and operating cost.

5. ACKNOWLEDGEMENTS

The DGIS project is a joint effort with the Office of the Secretary of Defense, Washington Headquarters Services, Directorate of Computer and Office Automation Resources. Project support for end user searching was provided by the Department of the Army, Office of the Deputy Chief of Staff for Research, Development and Acquisition. The Naval Surface Weapons Center, Technical Library, and the Air Force Environmental Technical Applications Center, Air Weather Service Technical Library, have provided support for the bibliographic post-processing efforts.

6. REFERENCES

1. Survey conducted in FY 1983.
2. Cotter, G.A., An Intelligent Gateway for the Department of Defense: The Technology Information System. June 1984, ADA-133-800, National Technical Information Service, Springfield, VA.
3. Hampel, V.E., Bailey, C., Kawin, R.A., "TIS" An Intelligent Gateway Computer for Information and Modeling Networks. August 1983, ADA-135-916, National Technical Information Center, Springfield, VA.
4. Jacobson, C.E., Cotter, G.A., The DoD Gateway Information System (DGIS) Directory of Resources. February 1986, Defense Technical Information Center, Alexandria, VA.
5. Jacobson, C.E., Cohen, R.S., Michel, J.T., Directory of DoD R&D Data Bases. September 1984, ADB-085-600, Defense Technical Information Center, Alexandria, VA.
6. Chastain, G.C., A Study of User-Defined Searching Requirements for the On-Line Version of the Directory of DoD-Sponsored R&D Data Bases on the Defense Gateway Computer System. March 1985, ADA-153-000, National Technical Information Service, Springfield, VA.
7. Williams, M.E., "Highlights of the Online Database Field - Gateways, Front Ends and Intermediary Systems," National Online Meeting Proceedings. May 1985, Learned Information, Inc., Medford, NJ.

8. Cotter, G.A., The DoD Gateway Information System. October 1985, ADA-161-701, National Technical Information Service, Springfield, VA.

9. Burton, H.D., The Intelligent Gateway: A Dynamic Resource Environment. January 1986, Lawrence Livermore National Laboratory, Livermore, CA.

10. Kuhn, A.D., Cotter, G.A., The DoD Gateway Information System (DGIS): Interface Design. March 1986, Defense Technical Information Center, Alexandria, VA.

AUTHOR AFFILIATION: Gladys Cotter is Chief of the Information Systems Division of the Office of Information Systems and Technology, Defense Technical Information Center, Alexandria, VA.

ADDRESS: Information Systems Division
Defense Technical Information Center, DTIC-EB
Cameron Station
Alexandria, VA 22304-6145

THE EVOLUTION OF GATEWAY TECHNOLOGY

Judith M. Hushon and Thomas J. Conry

Gateway technology incorporates the ability for computers to link together. The systems that are part of the gateway can be a permanently linked network of mini- and mainframe computers, as is the case with ARPANET. However, the more interesting case involves the interconnecting of an increasingly heterogeneous array of computers using a variety of links, including commercial value added networks (e.g., Telenet, Tymnet) and ordinary telephone lines on an as needed basis. As a result, the concept of a gateway has also come to mean linking of one computer to a variety of other remote computers. The earliest examples of this type of technology occurred in the late 1970's when both the Chemical Substances Information Network (CSIN) and the Technology Information System (TIS) were developed. These were large, multi-user gateway systems. CSIN was also among the first to provide the ability to search several different systems using pre-stored search strategies. EasyNet is also an example of a multi-user, gateway searching system.

Powerful desk-top micros can now handle a variety of connection protocols and are capable of providing a common interface for searching a wide variety of systems. While a large number of products facilitate connection to a single computer, only five software systems (MICRO-CSIN, Searchmaster, PC/NetLink, Sci-Mate and ProSearch) support not only accessing and downloading from multiple remote systems but also search systems using a single, pre-stored search strategy. The capabilities and limitations of these systems are compared.

Finally, future trends in gateway technology development are discussed, including: menu driven systems, common command language/translations, post-processing, accounting, and AI influence in database selection.

Gateways, Gateway Software, Microcomputers, Multi-User

The Evolution of Gateway Technology

Judith M. Hushon and Thomas J. Conry

The combination of computing and communicating is called networking. Computer networks link geographically dispersed information resources. Unfortunately, computer networks have been built in a fragmented way using an often bewildering array of technologies and protocols. As a result, many are incompatible. Gateway software facilitates this interconnection process. The term "gateway" has been broadly used and covers a variety of software that masks the incompatibilities of the networked systems to varying degrees.

Computer Wide-Area Networks

First, a short overview of computer networks is in order. In 1969, DARPA funded an R&D project that led to the development of ARPANET, the first wide-area network. ARPANET served as a testbed for development of advanced network protocols, including the TCP-IP protocol. This protocol, introduced in 1981, permits networks employing different technologies and connect protocols to be linked together while providing a uniform addressing scheme. Other major scientific networks include CSNET (1980), BITNET (1981), MFENET (mid 1970's), SPAN, DECNET, NCAR, RVAC, etc. In addition, there are publicly available networks such as TELENET, TYMNET and UNINET. These networks now link a wide variety of heterogeneous machines and operating systems.

Today, networks place a huge variety of remote computers within relatively easy reach of users, provided they know the varying protocols. Gateway software can be used to manage, simplify, and speed the connection to remote resources. This gateway software can either reside on the user's workstation or on the remote target computer.

The types of gateway software will be considered from two points of view: first, with regard to search and retrieval, and second, with regard to post-processing. Major examples of each category will be considered in this paper. Derivative systems such as DGIS, the DTIC customized version of TIS, and user-adapted EASYNET interfaces will not be specifically covered.

Categorizing of Gateway Systems

The vocabulary in the area of gateway systems is not highly developed. In order to clarify any discussion of this area, a more precise set of definitions is required.

Search and Retrieval

In the area of bibliographic searching and retrieval, the available set of gateway software can be divided into four increasingly complex and comprehensive categories. The first category is devoted to "basic gateway systems." This software undertakes the management of the telecommunications interface and logging into the target vendor system. LLNL's TIS presently

operates at this level as can such commercial products as Microstuf's CROSSTALK and Microsoft's Access.

The next level of software systems permits the user to pre-store a search strategy as a script which is then executed upon request, in addition to handling the telecommunications and login. These systems are therefore called "scripted gateway systems." The script is written in the native language of the target system and the script preparer must be accomplished at searching the target system and databases. Searchmaster, Insearch, PC/Net-Link, Wilsearch and DialogLink are examples of this scripted type of gateway system.

The next level of sophistication is embodied in systems that provide for translation from one vendor search language to another. All script preparation may still be in a vendor command language or in a common command language, but the software is present to carry out a general vendor-to-vendor translation. Pro-Search, for example, requires the user to use the Dialog or BRS command language and syntax while Sci-Mate uses a common, menu-driven command language. This opens up new searching possibilities because an end user no longer needs to know the command language of the vendor system, only a command language known to the software.

The highest level of searching sophistication presently available is embodied in the "expert gateway systems." These systems have a built in database of information on specific databases that lets them optimize their searches and have a set of rules that provide for vendor to vendor command translation. They know, for example, which databases support adjacency and which are indexed by CAS registry numbers. The term "expert" was not lightly chosen, for there are distinct parallels to what the artificial intelligence world is referring to as expert systems. Expert systems consist of a knowledge base, a set of rules (inference engine), and a user interface. These parts are all present in the expert gateway systems. Micro-CSIN is an example of this type of system. It even provides pre-stored lists of search terms tailored to optimize a search in that subject area across a number of databases.

Post-Processing

A separate set of categories are required to define the post-processing alternatives offered by gateway software. The "simple systems" offer no post-processing of downloaded citations. The next highest level of post-processing is provided by "standard systems" that support the conversion of records from variety of vendor systems and databases to a common format.

The "bibliography production systems" provide merging and sorting and may additionally include user specific formatting and the elimination of duplicates. Several systems have reached this level of sophistication, but only Micro-CSIN and TIS support duplicates elimination.

The final category of post-processing is denoted as "citation analysis." These systems support search term evaluation and frequency assessment. They may also incorporate a notion of relevancy assessment. LLNL's TIS is the primary representative in this category.

Multi-User vs Single User

Another way of looking at gateway systems is by dividing them into large centralized multi-user systems and those that operate exclusively on microcomputers.

Multi-User Gateways

The multi-user systems were, with a single exception, developed in the 70's when large machines were the only way to ensure sufficient computing power. The two primary examples are TIS at LLNL and CSIN developed with CEQ funding. Both of these systems were developed on DEC minicomputers running the UNIX operating system, CSIN provided search storage and simple search translation from a common command language while TIS only provides the basic gateway functionality. CSIN was also menu driven while TIS

is command driven. TIS' major power is derived from its post-processing functionality. Easy-net is a relative new-comer to the multi-user gateway scene. In sophistication, it is closer to CSIN in that it permits the user to enter a search strategy in a uniform, menu-driven manner and translates that strategy to the proper format for the identified vendor. The supported search logic is, however, less highly developed than with CSIN. It should be noted, however, that CSIN is no longer available and funding was diverted during 1985 to the development of a similar technology in a microcomputer environment.

Single-User Gateways

The other gateway software that will be discussed here is exclusively microcomputer oriented. The increasing power of microcomputers and their availability as well as their decreasing price has meant that most current gateway software development has been in this arena. There are a number of gateway software products that have been developed as front-ends to a single vendor's system. Often this software development has been undertaken by the vendor or the marketer of one or more databases to increase ease of use of the database system. Some examples of this type of bibliographic searching software are In-Search, DialogLink, Search Helper, Searchware, and Wilsearch. The first four search Dialog databases only and the last searches the Wilsonline files. (See Table 3).

Also available are five software packages that have the ability to search multiple vendor systems. These are: Micro-CSIN, PC/Net-Link, Pro-Search, Searchmaster, and Sci-Mate. These systems, however, differ markedly in their capabilities.

The main differences among the multiple system search software relate to:

- number of vendor systems interfaced
- whether the searcher must compose the search strategy in the host system's command language
- whether the software provides for translation of a query into one or more additional system command languages and the extent of this translation
- . whether the system provides assistance selecting databases and the extent of this assistance
- . whether the system provides session and monthly accounting data
- . level of post-processing
- . hardware requirements.

As can be seen from Table 4, the number and identities of the systems interfaced by the different software packages differ markedly. PC/Net-Link offers the broadest range of system accesses, but offers the lowest level of user support. Micro-CSIN and Sci-Mate probably offer the best mix of coverage and support. If a user is mainly interested in a Dialog interface, Pro-Search may be the most appropriate.

Of the five multiple system interfaces reviewed, only Micro-CSIN and Sci-Mate are truly vendor system command language independent. Pro-Search has the ability to use Dialog's command language to search on ERS and visa-versa, but the user must still be familiar with one of the two systems.

Only some of the systems offer assistance with the selection of databases and the level and type of assistance varies as well. Pro-Search, Micro-CSIN, and Sci-Mate provide database descriptions organized in a subject area hierarchy. Sci-Mate only provides this for a few of the most popular databases because of the file storage requirements for these extensive database descriptions. The user can read these descriptions and use them as the basis for selecting the databases to search. PC/Net-Link, on the other hand, asks the user to select a category to search and then selects the database for the user. In this way, PC/Net-Link can select certain cheaper systems in preference to other more expensive ones. The user cannot override the system's database choice.

One of the features being added to these multisystem interface software packages is the ability to capture and store accounting information from individual sessions and provide session and monthly summaries. At the present time, only Pro-Search and Micro-CSIN offer this feature but this may be one of the next additions to PC/Net-Link.

All of the software runs on IBM-PCs, though Micro-CSIN and PC/Net-Link require at least an XT with a 10 Megabyte hard disk. The systems also differ in their floppy disk drive requirements and their RAM requirements. Sci-Mate is the only package that has been ported to a variety of microcomputers; it has the ability to interface both DOS and CP/M systems. Depending on a user's hardware, Sci-Mate may provide the only available multisystem interface. The packages are also evenly divided according to modem support: some only support the Hayes Smartmodems (e.g., Searchmaster and PC/Net-Link) while others support an array of modem types (e.g., Pro-Search, Micro-CSIN, and Sci-Mate).

All of the discussed software packages are designed exclusively for bibliographic searching. Also, although each of the four multi-system interface packages provides the ability to store search strategies, only Micro-CSIN provides pre-tested lists of query terms organized by subject category that users can adopt as is or alter to meet their own needs.

Micro-CSIN is also the only system to incorporate a general file transfer capability. Kermit (developed and maintained by Columbia University) is distributed free of charge with Micro-CSIN and facilitates uploading and downloading of files in an error checked environment.

Micro-CSIN additionally includes the ability to obtain chemical identification information from the chemical dictionary databases on the vendor systems it interfaces. The inclusion of this information in subsequent bibliographic searches can greatly improve their accuracy and efficiency.

Post-processing of downloaded bibliographic citations is extremely useful, for it permits the automatic generation of bibliographies. Both Sci-Mate and Micro-CSIN have this capability. Sci-Mate uses the Personal Text Manager software and The Sci-Mate Editor to parse, sort, and edit citations and prepare bibliographies. Micro-CSIN can merge citations which also results in an elimination of exact duplicates and then sort them. In addition, it can use the RS/1 (TM) software to create formatted bibliographies. It should be pointed out that the TIS multi-user system also provides significant post-processing tools.

Future Directions

This discussion can serve to demonstrate the direction in which the multi-system, microcomputer, gateway software is evolving. The software is becoming more menu driven and user friendly. However, the most advanced systems have a way of by-passing the menus for the experienced user. In addition, there is a trend toward providing a common command language and automatic translation of the search request and system responses. This means that both novice and experienced users can successfully search databases on

a wide variety of vendor systems. Post-processing and accounting are other areas where the more sophisticated gateway software is evolving to better meet the needs of the user.

One final area where more research can be expected is in database selection. The selection procedure used by PC/Net-Link is too simplistic for broad application. It is designed to save money because searches are pre-paid and retrievals limited. However an expert system involving some inputs from the user concerning search preferences plus knowledge of database by subject coverage, vendor cost, time-of-day, vendor speed, etc. could speed-up database selection, especially for the amateur searcher. The user should, however, have the ability to override the system recommendations and make personal selections.

Other possible areas of development might include the ability for the system to evaluate the keywords or descriptor terms assigned by an indexer to a set of retrieved records. These keywords could be ordered by number of citations containing that index term and the user could be given the opportunity to select additional terms for inclusion in a modified search strategy and the search could be re-run or run against other databases.

Another related area where expert systems can be expected to make a difference is in determining the relevance of a set of citations. The relevance of a citation would be a function of the number of a selected list of terms that were applied to that

source by the indexers and abstracters. Weights could also be assigned to specific terms.

Gateway software is, therefore, still evolving. The trend seems to be not only toward greater functionality but also toward increased user friendliness. This will carry with it a requirement for increasing levels of microcomputer sophistication and power, but probably not increased hardware costs. As the number of "sophisticated" microcomputers grows, so will the demand for gateway software for more people will become searchers.

Table 1

Classification of Gateway Software Based on
Increasing Searching Capabilities

Category	Examples
Basic	TIS
Scripted	In-Search Wilsearch Searchmaster DialogLink PC/Net-Link
Simple Translation	CSIN Easy-Net Pro-Search Sci-Mate Searchware Search Helper
Expert Translation	Micro-CSIN

Table 2

Classification of Gateway Software Based on
Increasing Post-Processing Capabilities

Category	Examples
No Post-Processing	All Others
Post-Processing to a Standard Form	
Sort/Merge/Format	Sci-Mate
Duplication Elimination	Micro-CSIN
Citation Analysis	TIS

Future Trends

- Menu driven
- Expert modes
- Post-processing
- Accounting
- Database selection
- Search strategy refinement
- Citation relevance

Search Software Differences

- number of vendor systems interfaced
- whether the searcher must compose the search strategy in the host system's command language
- whether the software provides for translation of a query into one or more additional system command languages and the extent of this translation
- whether the system provides assistance selecting databases and the extent of this assistance
- whether the system provides session and monthly accounting data
- hardware requirements.

Table 3 Single System Interfaces

Capabilities	In-Search Menlo Corp.	Search Helper Info. Access	MicroDISCLOSURE Disclosure	Searchware Searchware	DunsPlus DunsPlus	DialogLINK Dialog	Wilsearch HW Wilson
Vendor systems interfaced	Dialog	Dialog	Dialog	Dialog	D&B	Dialog	Wilsonline
No. of databases interfaced	107	7	1	(all)	7	(all)	9
Autodial and login	yes	yes	yes	yes	yes	yes	yes
Offline search strategy preparation	yes	yes	no	yes	yes	yes	yes
Ability to store search strategy	yes	no	no	no	yes	yes	yes(one)
Ability to extend search online	yes	no	yes	yes	yes	yes	yes
Ability to search in native or script mode	no	no	no	yes	yes	no	yes
Menu driven	yes	yes	yes	yes	yes	yes	yes
Need to know host command language	yes	no	no	no	no	yes	yes
Provides translation	no	yes	no	yes	yes	no	no
Assistance with database selection	yes	yes	no	yes	no	no	yes
Accounting	no	yes	yes	yes	no	yes	no
Ability to retrieve records	yes	yes	yes	yes	yes	yes	yes
Post Processing	no	no	yes	no	yes	no	no
Audience	end users	info spec	end users	end users	end users	info spec	info spec
Hardware	IBM-PC, 192K 2 disk drive	IBM-PC, 128K 1 disk drive	IBM-PC, 128K, 2 disk drives	IBM-PC, 128K Apple II 1 disk drive	IBM-PC, 256K hard disk	IBM-PC, 256K Compaq 2 disk drives	IBM-PC, 256K IBM-PC, 256K 1 disk drive
Modems supported	most	Hayes Smart	Hayes Smart	most	most	most	Hayes Smart

Table 4 Multiple System Interfaces Including Micro-CSIN

Capabilities	Pro-Search Menlo Corp	Searchmaster SDC	Sci-Mate ISI	PC/Netlink Informatics	Micro-CSIN BBN
Vendor systems interfaced	Dialog 2, BRS	Dialog, BRS, NLM, SDC	Dialog, BRS, NLM, SDC, Questel, ISI	Dialog, BRS, NLM, SDC, Dow Jones, NEXIS, LEXIS, NewsNet, Questel, Westlaw, OCLC, Net-Search,	Dialog 2, BRS, NLM, SDC, OHS, Toxnet, CAS, CIS
Autodial and login	yes	yes	yes	yes	yes
Ability for user to reprogram existing host interface	no	no	no	no	yes
Ability to add user specified system interface	no	no	no	no	yes
Offline search strategy preparation	yes	yes	yes	yes	yes
Ability to store search strategy	yes	yes	yes	yes	yes
Ability to sequentially search multiple databases/hosts	no	no	no	no	yes
Ability to leave and return to script mode	no	no	no	no	yes
Ability to extend search online	yes	yes	yes	yes	yes
Ability to search in native or script mode	yes	yes	yes	yes	yes
Provides assistance in native mode	no	no	no	no	yes
Menu driven	yes	yes	yes	yes	yes
Ability to search factual/numeric as well as bibliographic databases in script	no	no	no	no	yes
Need to know host command language	yes (one)	yes	no	yes	no
Provides lists of query terms in a variety of subject areas	no	no	no	no	yes

Table 4 Multiple System Interfaces Including Micro-CSIN

Capabilities	Pro-Search	Searchmaster	Sci-Mate	PC/NetLink	Micro-CSIN
Provides translation	yes	no	yes	no	yes
Provides database specific translation	no	no	no	no	yes
Assistance with database selection	yes	no	limited	yes	yes
Ability to store database lists	no	no	no	no	yes
Vendor interaction displayed	no	yes	no	yes	yes
Accounting	yes	no	no	no	yes
Ability to retrieve records	yes	yes	yes	yes	yes
Post Processing	no	no	yes	no	limited
File maintenance capabilities	no	no	yes	no	yes
Ability to transfer files between PC workstations	no	no	no	no	yes
Audience	info spec	info spec	end users	info spec	end users
Hardware	IBM-PC, 192K 2 disk drives	IBM-PC, 128K 1 disk drive	IBM-PC, Vector, TRS-80, Kaypro, Apple II	IBM-PC/XT 10M hard disk	IBM-PC/XT or AT 10M hard disk
Modems supported	most	Hayes Smart	most	Hayes Smart	most

AUTHOR AFFILIATION: Tom Conry is a Microcomputer Information Specialist
for BBN Laboratories, Inc.

Judy Hushon was Manager, Washington Office, Research
Systems Department, BBN Laboratories, Inc. at the
time of this conference.

ADDRESS: BBN Laboratories, Inc.
1300 North 17th Street
Arlington, VA 22209

PRODUCT: INTELLECT

PRODUCT DESCRIPTION: No Product Description provided.

KEYWORDS: INTELLECT

VENDOR REPRESENTATIVE:

VENDOR: Artificial Intelligence Corp.

VENDOR ADDRESS: 100 Fifth Avenue
Waltham, MA 02254

PRODUCT: CAS ONLINE

PRODUCT DESCRIPTION: CAS ONLINE is a commercial online information service offering access to the chemical information files of Chemicals Abstracts Service, a division of the American Chemical Society. The user interface was designed to allow easy access by information professionals, while at the same time, supporting the infrequent user at his or her own level of competence. Unlike simple novice-expert systems, which address only the extremes of user competence, the CAS ONLINE user interface allows each user to seek his or her own level of communication with the service. This presentation will describe the CAS ONLINE user interface.

KEYWORDS: CAS ONLINE, Interfaces, Online Searching

VENDOR REPRESENTATIVE: Harry F. Boyle

VENDOR: Chemical Abstracts Service

VENDOR ADDRESS: P. O. Box 3012
Columbus, OH 43210

PRODUCT: DIALOGLINK and the DIALOG Business Connection

PRODUCT DESCRIPTION: This presentation will show two new DIALOG products developed for the purpose of simplifying access to DIALOG's online offerings. DIALOGLINK, DIALOG's communication and accounting software, provides such features as creation and editing of search strategies before going online; the ability to save, redisplay, and print information already received; local help functions; the ability to recall and modify the last command sent; and accounting features which allow the searcher to easily create detailed listings of search costs by searcher, client name, charge code, etc. The DIALOG Business Connection is a menu-based package designed for business professionals who have recurring needs for information on companies, products, and markets. It combines DIALOG's powerful search capabilities with the ease of use of an intelligent front-end.

KEYWORDS: DIALOGLINK, DIALOG Business Connection, DIALOG, Front-End, Menus, Online Searching

VENDOR REPRESENTATIVE: Randolph E. Hock

VENDOR: DIALOG Information Services, Inc.

VENDOR ADDRESS: 5 Cambridge Center
Cambridge, MA 02142

PRODUCT: MicroDisclosure

PRODUCT DESCRIPTION Database Contents. Contains annual and quarterly income statements, balance sheets, five-year summaries, company resumes, lists of officers and directors, subsidiaries, stock ownership, summaries for 5% institution and inside owners, ratio analysis, descriptive and SEC filing information for the New York Stock Exchange (NYSE), the American Exchange (AMEX), and over 6,000 over-the-counter companies.

Software: MicroDisclosure software provides online access and downloading capabilities for the Disclosure Online Database in a menu-driven environment. Other software features include searching and screening on 85 categories and items and preformatted or custom report generation. Data files can be converted into DIF format for use in Visicalc, Lotus 1-2-3, and other software packages.

Database Size: 10,000+ companies

Data Frequency: Quarterly, annual, and additional.

Time Span: For MicroDisclosure, up to three years for annual income statements, three quarters for quarterly income statements, and two years for balance sheets.

Updating: Weekly

Source: Company annual and periodic reports filed with the Securities and Exchange Commission, NYSE, and AMEX.

Hardware Required: IBM PC/XT or compatible, 128K memory, (192K with DOS 2.0 or higher), two disk drives or XT; 300/1200 baud modem.

KEYWORDS: MicroDisclosure, Disclosure, Menus, Online Searching

VENDOR REPRESENTATIVE: Diane Hoffman

VENDOR: Disclosure Information Group, Inc.

VENDOR ADDRESS: 5161 River Road
Bethesda, MD 20816

PRODUCT: WILSEARCH

PRODUCT DESCRIPTION: The H. W. Wilson Company developed a new and user-friendly microcomputer software product called WILSEARCH. It was designed for two major purposes: (1) To simplify online searching, to make it possible for a library patron to carry out online searches, without any previous training. (2) To make online searching affordable to the library patron by shifting time-consuming activities from the mainframe to the microcomputer. WILSEARCH is a menu-driven front-end system. Database selection may be left to the computer by specifying broad search categories. The software has been designed to select appropriate database(s) of his/her choice. WILSEARCH allows searching by author, title, subject, organization, journal title, date or range of dates, or classification number. It is possible to do Boolean searching (AND and OR), to use truncation symbols, and to automatically repeat the same search on different databases. When the searcher has developed a strategy by filling out the search screen, the system automatically logs on, searches, shows the results and downloads the records to the disk on the personal computer. Following downloading, the system automatically logs off, and the patron has an opportunity to review the downloaded records on the personal computer and can decide whether or not to print any of the records.

KEYWORDS: WILSONLINE, WILSEARCH, Front-End, Online Searching, Bibliographic Software

VENDOR REPRESENTATIVE: Bill Bartenbach

VENDOR: H. W. Wilson Company

VENDOR ADDRESS: 950 University Avenue
Bronx, NY 10452

PRODUCT: Sci-Mate

PRODUCT DESCRIPTION: The Sci-Mate Software System consists of three components for information retrieval and management. Each works alone or in combination with one or two other components. All three are menu-driven.

The Sci-Mate Searcher provides automatic log on for databases on BRS, DIALOG, NLM, ORBIT, and QUESTEL. It translates search terms and simple menu choices into the specialized search language of each host. It enables users to create search strategies offline and to download database records. Records may be exported to the Sci-Mate Manager and Sci-Mate Editor.

The Sci-Mate Manager is a file management program for the storage and retrieval of records downloaded from online databases or entered at the keyboard. Records may be edited, searched, and sorted. Other features include a label generator and a report generator.

The Sci-Mate Editor formats bibliographic references from records downloaded with the Searcher, imported from the Manager, or entered at the keyboard. Fifteen style sheets are supplied with the program; users may also design their own. The Editor offers five sorting options, including cited-order sorting. It also embeds full or abbreviated references in a manuscript prepared with a word processor.

KEYWORDS: Sci-Mate, Online Mediation, Gateway, Front-End, Information Retrieval System, Microcomputers, File Management, Text Processing, Bibliographic Software, Menus

VENDOR REPRESENTATIVE: David E. Toliver

VENDOR: Institute for Scientific Information

VENDOR ADDRESS: Software Products
3501 Market Street
Philadelphia, PA 19104

PRODUCT: SIRE

PRODUCT DESCRIPTION SIRE is an information retrieval system that offers automatic full-text indexing on up to 256 fields per document, with no limits on the lengths of fields or documents. Queries can be in natural language, and may also utilize full Boolean logic (including parentheses and reusing previous searches), proximity, field restrictions, partial matches, and numeric ranges. Words are automatically matched on their roots. Retrieved documents are ranked according to their likely usefulness. Documents may be included in a query to look for similar documents. A word or phrase can be expanded to show related words; this is done automatically by statistical methods, without the user building a thesaurus.

SIRE's enhanced search features are available to the sophisticated searcher, but do not intrude upon the casual or new searcher. A user of SIRE can search and display documents without knowing any commands. Most searches take just a few seconds.

SIRE runs on a variety of hardware and operating systems, including MS-DOS, UNIX, and VMS.

SIRE is a trademark of KNM, Inc.

KEYWORDS: SIRE, Information Retrieval System, Microcomputers, Automatic Indexing, Natural Language Processing

VENDOR REPRESENTATIVE: Matthew B. Koll

VENDOR: KNM, Inc.

VENDOR ADDRESS: 6118 Swansen Street
Bethesda, MD 20817

PRODUCT: EasyNet

PRODUCT DESCRIPTION: No Product Description provided.

KEYWORDS: EasyNet

VENDOR REPRESENTATIVE: Marvin Weinberger

VENDOR: Telebase Systems Inc.

VENDOR ADDRESS: 134 North Narberth
Narberth, PA 19072

PRODUCT IRVING Library Network

PRODUCT DESCRIPTION. The IRVING Library Network has drawn national and world-wide attention in the library community as one of the most significant efforts to link dissimilar library computer systems. IRVING allows patrons and staff to obtain bibliographic data from different systems by providing them with the same set of easy-to-use screens regardless of which kind of computer system is being searched. The Network has been in successful operation in Colorado since July of 1985 and is currently being expanded to link the Denver, Boulder, Jefferson County and Aurora Public Libraries.

 The IRVING Network uses DEC MicroVAX II computers as network processors to interact with each library's bibliographic system by emulating the system's unique search commands. (A single network processor can be shared by several computer systems.) IRVING requires no changes to existing bibliographic system hardware, software, or databases; the security of the existing systems is completely maintained.

 The IRVING Network makes use of the Open Systems Interconnection (OSI) model and X.25 packet switching. The IRVING architecture can be applied in a number of settings which require a standard means of obtaining data from a variety of different database systems.

KEYWORDS: Library Automation, Database Management Systems, Bibliographic Searches, Open Systems Interconnection, Standard Interfaces, X.25 Packet Switching, Common Command Language

VENDOR REPRESENTATIVE: Jim McHeyser

VENDOR: Minicomputer Systems Incorporated (MSI)

VENDOR ADDRESS: 2037 16th Street
 Boulder, CO 80302

Issues for Expert Systems for Retrieval Assistance

Richard S. Marcus

The prospects for expert computer systems that would match or excel human experts in providing assistance to users of retrieval systems have been investigated in a number of recent efforts. In this paper we consider the issues in developing such systems. In particular, we contrast such efforts in terms of the models for retrieval and assistance they subsume and in terms of the techniques for performing effective retrieval and for developing expert assistance systems. Further, we state and attempt to support three premises: (1) to provide a truly comprehensive expert retrieval assistant requires a very extensive knowledge-base development; (2) there are significant questions concerning retrieval models and assistance techniques which need to be resolved in developing such expert systems; and (3) although expert retrieval assistance development is difficult, it shows promise for deepening our understanding of the retrieval process from a basic scientific viewpoint as well as for improving search techniques themselves. In support of these premises we discuss some of our recent experiences in the development of our CONIT experimental retrieval assistance system. Other issues we have considered in our research which are illustrative of these points are listed below and discussed in the paper: How does one successfully integrate a menu mode for maximum assistance with a command mode for maximum flexibility and efficiency? How much background problem exposition needs to be drawn from the user? In what form? How is it used for assistance? How does one capture the underlying conceptual formulation of a user's problem and distinguish it from specific search strategies without confusing the (impatient to finish the search) user? How does one develop "operators" which take existing search strategy and modify it in a given way? What are the best types of search strategy to espouse in given contexts?

Expert System, Retrieval Assistance, Search Models, Search Strategy,
Search Modification Operators

AUTHOR AFFILIATION: Richard Marcus is Principal Research
Scientist at the Massachusetts Institute
of Technology Laboratory for Information
and Decision Systems.

ADDRESS: MIT
Laboratory for Information and Decision
Systems
Room 35-414
Cambridge, MA 02130

OAK -- A NEW APPROACH TO USER SEARCH ASSISTANCE

Charles T. Meadow, David B. Macdonald
Sue M. Ewing and Deborah Dowson

The OAK System (Online Access to Knowledge) consists of tutorials and user assistance software that enable users to search online databases at the U.S. Department of Energy's Technical Information Center. OAK combines menus with other conversational methods to remove completely the need for users to learn a new command language in order to search. The system offers two computer assisted instruction courses, one covering database searching in a broad introduction and the second describing how a search is conducted using OAK. Search assistance is provided with initial query formulation, browsing and analysis of retrieved records, and in deciding how to modify an earlier search formulation. A basic design principle has been that the user maintains control over the search at all times, although he or she may request suggestions from OAK on how to proceed. A study of users established that they are both well educated and well motivated to perform searches on their own.

Online Searching, Database Search, Gateways, User Assistance,
Front-End, Decision Assistance

1. INTRODUCTION

OAK -- Online Access to Knowledge -- is a software system designed to make database search services both easier and more attractive to end users. It is being developed by California Information Company for the U. S. Department of Energy for use in connection with DOE's arms control database.

Many end users would like to do some of their own searching. Many library staff members would like their patrons to be able to do at least the easy searches on their own. It is for this end user environment [Borgman et al, 1985] that we have designed the OAK system.

End users know their subject well and the literature moderately well, but tend to have no knowledge of or interest in the mechanics of searching not at all. They would prefer that the mechanical requirements to go away. To gain their interest, it is necessary to develop access languages that fits these users' characteristics [Meadow, 1981], not to seek users who can utilize the languages.

What, then, do we want a computer search intermediary to do? The first objective is to relieve the user of the need to learn an artificial, programming-like language and, instead, to enable him or her to communicate with a search service in meaningful terms. The second objective is to help assure quality of results. These are somewhat conflicting aims. There are intermediary systems that make the search process almost trivial, but do not accomplish the second objective. There are others that so emphasize potential achievement as to neglect the critical roles of relieving the user of technological tedium and assisting him or her to achieve results.

We can go further with these concepts. We believe the complex command languages used by most search services are antithetical to the objective of serving the end user. These languages may distract the user from the true objectives: defining, searching for and evaluating information. They force concentration on design of a process and tend to focus attention on syntactic detail instead of concept formulation and pattern recognition. It is not necessary that an intermediary be able to comprehend natural language, but it is necessary that the user be able to communicate in terms meaningful to himself. Mooers [1960] said long ago that if an information retrieval system is difficult to use, it will not be used. Put another way:

For a person using a computer as a problem solver, the perception of both the problem set that is appropriate to attempt and the acceptability of performance and solutions is governed by the problem-solving language of the user. Minimal language knowledge leads to minimal perception of possible solutions, and minimal perception of possible solutions leads to minimizing language learning. [Meadow, 1983]

There are other user intermediary systems for online search services. However, we believe our approach is unique. OAK is not and cannot be a single, general program, to fit all users and all database services. All front ends simplify the

language of searching, but OAK also helps the user, while leaving him in control. We intend that it be tailored to a user group/search service combination, so that it can appear to be as knowledgeable as possible about the subject matter and databases in use and as helpful as possible to the user. The wider the range of knowledge it is expected to have, the less valuable it is to most users.

2. GENERAL DESIGN APPROACH

The OAK system has two major components: tutorials and assistance programs. Tutorials are for instruction before the user begins to search. Assistance programs help in the actual conduct of a search. Since good instructional design often involves having students do the real thing in an instructional setting, the tutorials have elements of search assistance in them.

Oak is designed for IBM PC-compatible computers and a variety of modems and transmission speeds.

2.1. INSTRUCTION

The principal objective of the tutorials is to establish in the user's mind a mental image of the retrieval system and what he should expect when he tries to use it. The present version of OAK does not teach the use of a command language. Hence the tutorials do not teach commands, but functions; not mechanics but concepts.

The tutorials introduce the concepts of database and database search system. They teach what is expected of the searcher, in the way of defining the subject; that searches should not be expected to be perfectly executed the first time; and that iteration is an essential part of the process. Because we do not teach a specific command language, users completing OAK tutorials will thoroughly understand the search concept, but will not have learned the mechanics. A printed user's guide is being developed to serve as a reference.

2.2. ASSISTANCE

The assistance programs provide some teaching as well. We have made the assumption that some people will try to search without having previously taken the tutorials or read the user's guide. There are extensive help options so that a user who wants to be lead, step by step through the procedure can be, but a more experienced user can work rapidly, without excessive interruption for elementary questions. We see no reason whatever to involve the user in typing unfamiliar symbols using a rigid syntax, when our software can easily do it for him with no loss of information.

It is central to the entire OAK approach that we do not attempt to supplant the user's ultimate primacy in knowing what he is looking for and in judging the results.

In our approach, the software performs the mechanical work and assists the user in making such decisions as how to proceed with the search, what terms to use in a search, and how to analyze

preliminary results. Since the user is present during the entire search, and we find it a considerable economy in system development, and an added factor in assurance of user satisfaction, if the user makes the intellectual decisions.

3. DESIGN

There are two primary ways to perform a search through OAK: (1) for the user to compose a search formulation and (2) for him to use an OAK script, or prepared search that requires at most that the user supply the values of some specific search terms. Both approaches stress user involvement in decisions and both try to keep the user's attention focused on goals, rather than on the mechanical details of searching.

3.1. USER-COMPOSED SEARCHES

Strategy Selection. An OAK search begins with an option to provide advice on selection of a broad approach to search strategy. Experienced users can by-pass this; first-time users, we hope, will ask for it. Roughly, we present the user with a choice of three strategies. Like many other actions we take in OAK, offering the choice is done as much to make the user think in these terms, the better to understand the results and make decisions on improvement, as it is for OAK to make direct use of the decision during the search. The choices are:

Known item: the user knows (approximately) about one document, author, source, etc. that can be used as the basis for beginning and then broadening a search.

Standard: the user knows several facets, or subject subdivisions, which can be used to compose a query. These can, of course, be revised later, if necessary.

Broad: the user intends to begin with a broad subject definition (or other broad facet, such as sponsoring agency) and then to gradually narrow the search down, as he sees the numbers of documents involved.

Search Formulation. Formulation of a search is a three-step process, always followed by an opportunity to review and revise what has been done.

(1) OAK begins by asking for a natural language statement of the search objective. This is done mainly to focus the user's attention on his own objective. It has been our experience in teaching searching that beginners often become so involved in the mechanics that they neglect to think consciously about their own objectives. OAK draws attention back to the objective and, as noted above, to the means of accomplishing it.

(2) The next step is to subdivide the main topic into facets or categories, such as subject, author, language or date. A classic although not infallible approach to problem solving, subdividing it into smaller ones is far more familiar to most professional people than is formulation of a subject into a

Boolean expression. OAK users are not involved in creating Boolean expressions; emphasis remains on problem definition rather than mathematical expression.

(3) Finally, for each facet or category, we elicit a list of terms to be used in actual searching. Now, we are asking for terms which will be the ones actually used in a search. Assistance in selecting terms, as described below, will be available if the user wants it. Essentially, the process is one of completing a form approximately as shown in Figures 1 and 2.

Upon OAK-assisted completion of this table, the search has been formulated. The Boolean logic implied is that all terms within a facet are conjoined, or connected with the OR operator. All facets are treated as sets of terms and all such sets are disjointed, or connected by AND.

In Boolean terms, the search shown in Figure 1 would be written:

```
(seismic detection + seismic events + seismic effects) *  
earthquakes * california * (DATE >= 1984)
```

For each possible category, a micro-script will assist the user in providing the individual terms in the appropriate form. If a category is author, then the script will elicit last name and first name or initials separately to avoid forcing the user to memorize the correct format.

Review and Revision. When the table is complete, the user is invited to review it, compare it with his or her own stated goal, and revise it if desired. Revision at this stage is a simple matter of adding, deleting or modifying terms or facets.

Translation and Transmission. When the user has approved the table form of his question, it is automatically translated into command language and transmitted to the search service. If the communications link to the search service has not yet been established, that step is taken at this point. All telecommunication functions are automatic and user transparent, once a file of communications parameters has been initialized. These steps are not seen by the user.

Decision Making. The most challenging part of a search comes now. A formulation has been made and some results achieved. The user must now judge whether these constitute an acceptable solution to his problem. Judgment is usually in terms of the number of records retrieved, relevance or value of the retrieved records in the user's view, and completeness of the results or what is technically called precision (ratio of wanted to unwanted records). But, the user is free to use any criteria he or she likes in making a judgment.

Having made a judgment, which need not be explicitly stated, a decision must be made as to what to do next. Almost all retrieval systems give a statement of how many records were retrieved as the immediate response to a search command. This is frequently a basis for decision, in itself [Cunningham and Weinberg, 1985]. The user will be given a list of what can be done next. One option is to ask OAK for decision-making assistance.

We will use a two-tiered decision tree, first asking for a

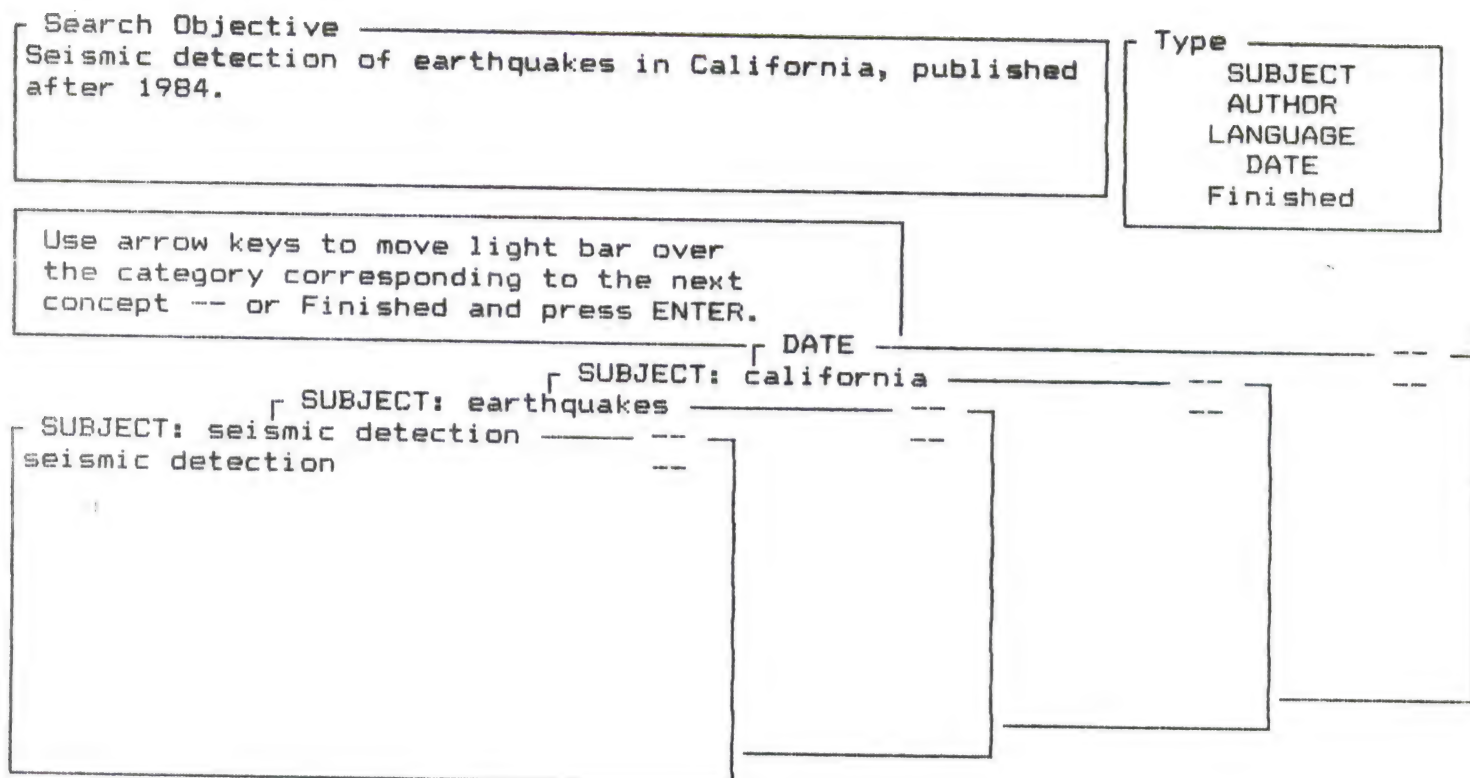


Figure 1. The configuration of an OAK search: the natural language search objective and the names of the major facets.

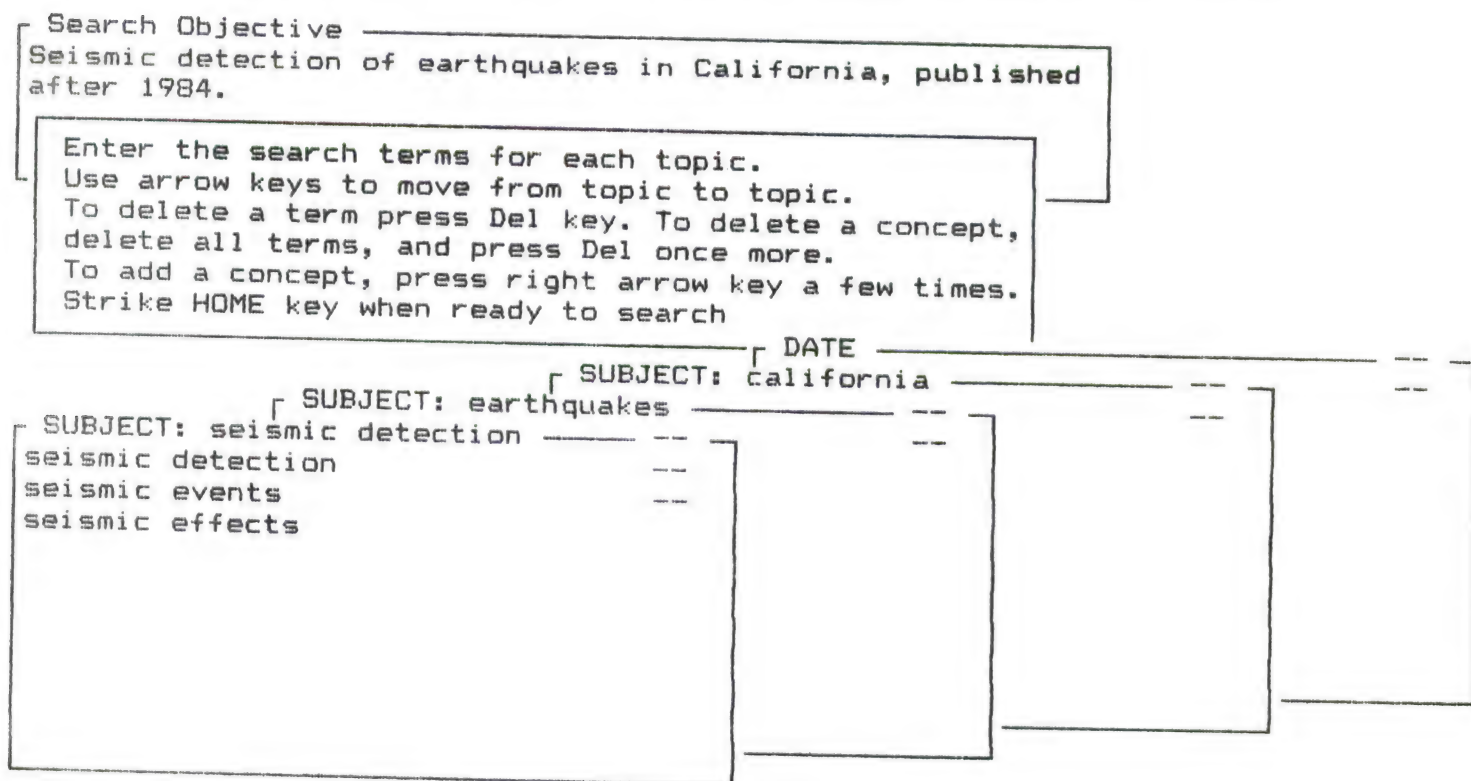


Figure 2. Additional terms have been added to the facets and instructions have appeared for additional search editing.

choice among

- Editing the search
- Starting a new search
- Browsing and analysis
- Quitting

and then having several options under each major heading selected.

Editing provides the opportunity to make changes in an existing search.

Starting a new search is always an option and, in a sense, is a major form of revision.

Analysis in the first version of OAK includes the following activities:

Browsing and evaluating individual records viewed. Unless zero records were retrieved, this is always our first recommendation for an action to take.

Set size analysis, in which OAK brings to the user's attention terms or facets which retrieved no results or excessively large numbers of records.

Term analysis, in which OAK gives another opportunity to look terms up in the dictionary to verify spelling or look for similar terms. (The first such opportunity comes while the query was being formulated.)

Quitting can imply satisfaction with results or the opposite. A satisfied user will be helped to record his results on a printer or disk, to store his search formulation for future re-use, and to use other software for subsequent processing (e.g. word processing for editing of results).

3.2. SCRIPT SEARCHES

In many applications of information retrieval users deal with a relatively small number of questions, which recur often. In such cases the search can be composed in advance, and no more is asked of the user than to supply the value of some data field at the time of the search. He has no need to formulate the logic of the complete search. This is done entirely in advance.

For example, a person may frequently need to search a bibliographic file for information on treatment of the toxic effects of He could perform such searches, supplying only the name of the substance of current interest at search time. Other examples, from other subject fields are: a search for material by any given author; a search for information about the interaction of two drugs in terms of patient effects; or the compilation of a profile of a particular company. In these cases, the user would need only supply the author's name, the drug names or the company name.

OAK will supply a small library of script searches and include the capability for users to compose their own scripts and

to store them in the library. A mini-script will be used in user-composed searches for each type of facet, such as subject term, author, date, etc.

The use of a script is, of course, a compromise between the extreme ease of searching this way and the lack of user control over the exact search logic. But, because standard searches can so often be used, scripts can be made available and the user given the option of using them or not.

4. DEVELOPMENT AND TESTING

The first operational version of OAK will be available for distribution to DOE users in the Spring of 1986. The software is being user-tested by our evaluation consultants at the University of California at Los Angeles.

In another sense, OAK can never be "completed." There will always be new ideas for improvement or extension, many of which will arise only from user feedback. Thus, we can only consider OAK to be a software system under continuous development.

5. Future

OAK has been designed with one guiding principle: that user assistance software should be as well matched as possible to (1) the kinds of people using it (their education, subject interest and computer familiarity), (2) the target system or database service in use, (3) the databases in use and (4) the nature of the problems likely to be solved.

In OAK develops, we prefer to see a number of similar systems, rather than a single, overly general one. Different user-target system combinations will lead to different forms and appearances of OAK. Our basic design and its variations are based on perceived problem-solving needs of users.

The major improvements we expect to see are:

Introduction of advanced analytic assistance procedures, such as (1) frequency analysis of terms in retrieved records, (2) discriminant analysis to find term values that differ in frequency between highly valued records and the complete retrieved set and (3) record clustering or association.

Linkage of OAK to post-processing software, to complete the processing of retrieved data.

Linking OAK to more than one database service at a time, to permit it to perform multifile searches in the true sense of that term.

We believe that, in a few years, this kind of software will become the universal way to deal with databases. Whether based on OAK or some other system, users will want and receive specialized assistance in dealing with database services and other complex information systems and the current craze for complex, programming-like command languages will quietly disappear.

REFERENCES

Borgman, C.L.; Case, D.O.; Meadow, C.T.; Incorporating users' information seeking styles into the design of an information retrieval interface. Proceedings of the Annual Meeting of the American Society for Information Science, 22: 324-330, White Plains NY: Knowledge Industry Publications Inc; 1985.

Cunningham, C.A.; Weinberg, B.H.; Search strategy in the field of engineering: a statistical approach. IN Williams, M.E.; Hogan, T.H., Eds., Sixth National Online Meeting, Medford NJ: Learned Information, Inc.; 1985.

Meadow, Charles T.; Matching users and user languages in information retrieval. Online Review 5 (4): 313-22, 1981.

Meadow, Charles T.; A Study of User Adaptation to an Interactive Information Retrieval System, OCLC Research Report OCLC/OPR/RR-83/6, Dublin OH, OCLC Online Library Center, Inc., 1983; p. 4.

Mooers, Calvin; Mooer's Law or why some retrieval systems are used and others are not. Editorial. American Documentation, 11 (3): 204, July 1960.

AUTHOR AFFILIATION: Charles Meadow is a Professor of Library and Information Science, University of Toronto and President of California Information Company.

ADDRESS: Department of Library and Information Science
University of Toronto
Claude T. Bissell Bldg.
140 St. George Street
Toronto, Ontario M5S1A1
Canada

WHETHER AND WHITHER MICRO-BASED FRONT ENDS?

David E. Toliver

There are four places at which front-end software can be inserted in the retrieval system process: the databank mainframe, a mainframe remote to both the databank and the user, a mainframe to which the user has direct access, and a personal computer directly available to the user. This paper summarizes the advantages and disadvantages of each. Special emphasis is made on the unique capabilities that the microcomputer brings to the front-end. In particular, the microcomputer is best suited for providing the user with a rich interactive environment that can graphically define, control, and present the progress of the search. Combined with AI methods that are becoming increasingly practical for microcomputers, the microcomputer solution to the front-end problem has a very promising future.

llk

Online Mediation, Gateways, Front-End, Information Retrieval,
Databanks, Microcomputers, Interfaces, Artificial Intelligence

1. Introduction

Front-end software can be placed in at least four locations in the retrieval system process: the databank mainframe, a mainframe with asynchronous telecommunication links to both the databank and the user, a mainframe to which the user has direct access, and a local personal computer. This paper will summarize the advantages and disadvantages of each. Special emphasis will be given to the unique capabilities that the microcomputer offers as a front-end, both as presently used and its potential for use in the future.

First, to clarify the terminology used in this paper, I will follow Pendleton and Hawkins in some of their definitions for this emerging field. [1-4] "Databanks" are the computers on which databases are located. "Telecommunications links", or in short, "telecom links", are the packet-switching networks that provide access to the databanks, typically Tymnet and Telenet. "Gateway" software automates access to databanks or databases, but does not translate or otherwise assist in the process of information retrieval. "Front-end" software automates access and further assists in the information retrieval process on databanks and databases.

2. Examples of front-end systems by placement

First, examples of front-ends placed on the databank mainframes include Colleague and After Dark, both services of BRS. BRS has recently tried to break through to the end-user market with BRKThru. Dialog competes with its Knowledge Index, an inexpensive after-hours front-end on the Dialog mainframe. They have also just introduced the Business Connection, a unified front-end for about a dozen business databases.

Second, examples of front-ends on mainframes linked to both users and databanks by asynchronous telecommunications include two large experimental systems and two commercial systems. The Multics mainframe at MIT allows dial-up access to the experimental CONIT front-end software. The Chemical Substances Information Network (CSIN) served as a switching host for medical and toxicological databases. EasyNet is a private service that promotes end-user searching by charging a fixed rate against credit cards, and then only if records are found for the user. A gateway system run by Telecom Canada, iNet 2000 serves as a dial-up switching service to 44 different databanks.

Third, locally accessed mainframes are the most unusual arrangement for front-end software. For the staff at MIT with direct connections to the Multics mainframe, CONIT may be cited as an example in the research community. IT from UserLink Communications is the only commercial front-end product that Hawkins, in his recent study, cited as running on a mainframe, in this case a VAX.

Finally, the largest category of front-ends are those products and packages that are made for microcomputers. They may be further classified as those front-ends which access multiple databanks, those which access a single databank, and those which access only a specific set of databases on one or more databanks.

Examples of micro-based front-ends that access multiple databanks and databases include: the Sci-Mate Searcher from ISI, one of the earliest entries; Pro-Search, developed by Menlo Corporation and now available from PBS; Search Works, a new entry from Online Research Corporation; and Search Master, a gateway package from SDC.

Examples of databank-specific micro-based front-ends come mainly from the databanks organizations themselves. These include Micro-CSIN for professional intermediaries and the Grateful Med for end-users, both from the National Library of Medicine. Dialog-link is a gateway program from Dialog.

Examples of database-specific micro-based front-ends come from a variety of database producers. These include microCAMBRIDGE for databases from Cambridge Scientific Abstracts, microDISCLOSURE for databases from Disclosure, Inc., Search Helper for databases from Information Access Co., and WILSEARCH for databases from the H.W. Wilson Company.

3. Features that all front-ends can share

Placement of the front-end significantly affects the features the front-end can offer. However, there are some capabilities that can be expected in a front-end which are not affected by its placement. These include the potential ability of front-end systems to:

(1) Simplify the use of databank(s) retrieval language by providing prompts, helps, hints, and assistance with the retrieval language through query analysis, translations, menus and expanded prompts;

(2) Simplify use of databases with fields presented in menus, explanation, examples, and field-edit controls; and

(3) Provide multi-database processing in which queries for one database are applied to another and terms retrieved from one database are turned into queries for another.

4. The front-end on the remote databank itself: pros and cons

The advantages of having the front-end running as a process directly on the databank include:

(1) A broad range of terminal devices can access the system, including older "dumb" terminals as well as any personal computer running one of the generic communications software packages.

(2) The closest coordination between the front-end developers and the databank maintenance group is possible. This allows front-end changes that accommodate retrieval language changes and database reloads to be made quickly and smoothly.

The disadvantages of having the front-end running on the databank mainframe include:

(1) The front-end will only mediate the databases on the databank itself. Different front-end interfaces and languages must be learned if more than one databank is used.

(2) Users must still manually access the databank by negotiating the modem, network, and databank protocols, or use separate PC telecom or gateway software to access the front-end.

(3) Screen refresh is limited to low-speed asynchronous baud rates, making only terse interactions economical.

(4) There is no opportunity to build derivative downloaded files and to modify and customize the results.

(5) All interaction with the front-end involves timed charges, in addition to those clocked by the retrieval system.

5. Remote front-end or gateway mainframes: pros and cons

The advantages of having the front-end running as a service on an independent dial-up mainframe include:

(1) Access is available to multiple databanks and databases, with a uniform language made possible by translation routines.

(2) Centralized billing for multiple databanks can be arranged at the gateway. EasyNet, for example, requires the use of a credit card.

(3) Any ordinary terminal or PC running telecom software can access the front-end mainframe.

(4) Qualified technicians are directly involved in timely installation and maintenance of upgrades, keeping this task transparent to the user.

The disadvantages of having the front-end running as a service on an independent mainframe include:

(1) Screen refresh is still limited to low-speed asynchronous baud rates, making only terse interactions economical.

(2) All activity with the front-end, including any pre-session and post-session processing, involves timed charges.

(3) The user must still perform access steps to the front-end switching mainframe, although a separate telecom program or smart terminal can perform this function.

6. Local mainframe with direct high-speed access: pros and cons

The advantages of having the front-end running as a program on a mainframe with a direct high-speed terminal link include many of those cited for a remote switching mainframe:

(1) Multiple databanks and databases can be accessed through a standard command interface.

(2) Centralized billing arrangements are possible.

(3) Qualified technicians are directly involved in the installation and maintenance of upgrades.

The direct high-speed terminal link on a direct-access mainframe provides additional advantages, including:

(1) A richer interactive environment is possible with the high-speed video interface, improving visualization of the process, potentially involving a graphic presentation.

(2) More information about the information system can be provided quickly and at lower cost while connected online.

(3) Interfaces are possible with other user packages, such as data management systems and word processors.

(4) Mass storage facilities for strategies and results, including the construction of derivative databases, can be shared by researchers working on the same project.

The disadvantages of having the front-end running as a program on a mainframe with a direct high-speed link include:

(1) Only one package in this category is currently commercially available: IT from the U.K., with a strong emphasis on access to European databanks.

(2) Relatively few users have access to a mainframe that will run the software. Thus, it is not suitable for organizations that cannot commit to shared computer resources.

(3) Software packages purchased or developed for such environments generally cost much more than software for microcomputers, given the higher development cost and the smaller customer base.

(4) A wide variety of video-control and print terminal standards may have to be accommodated. If a TTY terminal is assumed, the scope of interaction is still limited; if an advanced terminal is required, fewer users will be able to take advantage of the facility.

7. Front-ends on local microcomputers: pros and cons

The advantages of having the front-end running as a program on a local microcomputer include both technical and marketing factors. Technical factors include:

(1) The feature-rich high-speed video interface provides an interactive environment with improved visualization of the process including possibly a graphic representation.

(2) With a fast video interface, a great deal of information can be put onto the screen in very little time, minimizing the cost of support information while connected online.

(3) Preparing data for uploading and processing the results of downloading can be done locally without the pressure of any clocked charges.

(4) Mass-storage for strategies and data is inexpensive and involves no time-based storage costs.

(5) All access: modem, network, databank, and database, can be automated under the control of a single software package.

(6) Interfaces with other commonly used packages such as word processors and data managers can be established.

Marketing factors that favor the development of microcomputer-based front-ends are passed on as advantages to the user in terms of selection, options, and cost. These include:

(1) The large installed base of PC hardware (3-5 million IBM-PCs and clones) has attracted developers to the potential market for front-end systems.

(2) PC-DOS has defined a de facto standard for video control and interface conventions, with a broad and growing selection of programming productivity tools and turnkey run-time modules spurring rapid development.

(3) This has resulted in a large selection of packages filling a variety of front-end requirements that are reasonably priced as they compete for the users' dollars.

The disadvantages of having the front-end running as a program on a local microcomputer include:

(1) User may have to become directly involved in the installation, maintenance, and update activities.

(2) Distribution methods for updates are most complex. Only with micros do updates involve extensive production and distribution. The response by developers to changes in retrieval protocols is usually the slowest of all four configurations.

(3) The delay in response to databank and database changes can keep users from accessing resources for extended periods.

8. Emerging technologies for micro-based front-ends

Advances in microcomputer hardware and software development can be expected to significantly affect the development of micro-based front-end systems. Hardware advances include ever more powerful personal computers and their processors, e.g., the IBM-PC/AT and those based on the advanced 68020 and 80386 micro-processors. One megabit dynamic RAM chips are becoming available to support large memory-based programs.

High-density local mass storage is increasing with 100 Mbyte fixed disks now available for under \$2,000. Vertical recording technology implies increased density of at least an order of magnitude. For instance, one hardware manufacturer claims to be developing a floppy disk that can hold 10 megabytes of data!

The density of bit-mapped graphics is increasing. IBM's Enhanced Graphics Adaptor will be replaced by even more advanced devices that give almost photographic quality presentations. Mice, touch-screens, and other pointing devices provide new ways for humans to communicate to machines.

Software advances include new operating systems and operating environments such as Windows, TopView, and GEM; new compilers and development systems; new system tools and run-time packages such as asynchronous telecommunication managers, window managers, and screen and forms generators.

Many of these advances in microcomputer performance will make Artificial Intelligence processing practical for micro-based front-ends. Most important are the more powerful and faster processors and the larger internal and mass-storage memories. Although most advances today in AI are being made on larger machines, AI as a tool for the front-end application should migrate to microcomputers within the next five years.

The most significant advantages of the micro-based front-end today are its fast and feature-rich human input and output interfaces, along with the system software that has evolved to manage them. With the IBM-PC has emerged a de facto standard for

video control. Because of this standard, many generic tools and techniques have been developed that make it possible to develop applications quickly and painlessly. These interfaces can be applied in new and compelling ways to the front-end application.

9. Scenario for a future micro-based front-end

In this scenario, proposed uses of Artificial Intelligence will not be emphasized. Although AI should eventually play a central role in future front-end systems of every configuration, its present requirements are best met with mainframe-based front-ends. Instead, this scenario incorporates advances in interface design that can be applied now to front-end programs. These include new tools, capabilities, and methods available for human interfaces on the microcomputer.

Prior to accessing a databank, the user is assisted in constructing "concept pools." These are sets of keywords or phrases that make up a search strategy. OR-logic will eventually be applied to the members of these sets. The sets may begin with seed-words parsed out of a natural language search query. Then they may be elaborated in at least three ways: (1) by a local knowledge-base that maps the terminology in a discipline, (2) by a dialog that elicits synonyms and related terms, or (3) by a combination of these and other concept-identification methods.

The concept pools are presented as graphic regions on the screen: circles or ovals that encompass the terms in each pool. Prior to going online, the user is asked to rank-order the words within each pool, from the most to the least important. Words can be repositioned within the pool by using a mouse device to point, click, and drag the words into relative position. The user may at this time overlap pools to introduce AND-logic into the query.

The user may explicitly select a database or set of databases, or allow the software to help in doing so. The latter may involve the application of AI: analysis of the query and comparison to a knowledge-base about databases. Or database selection could be based on the answers to a set of discipline-classification questions. In any case, the negotiation of modem and network protocols is automated. Passwords may or may not be issued automatically.

The graphic representation of the query is dynamically transformed into the Boolean-logic commands required by the host. First, sets are created for the highest ranked terms, one at a time. If the results indicate very small numbers of hits, OR-logic is applied to subsequently ranked terms in the pool. When the results are of manageable size and the conjunction of the pools is likely to return results, AND-logic is applied.

AI may be used to determine the probabilities of getting hits and to make adjustments to the dynamically generated search strategy. Some of the ideas for monitoring the progress of a user's strategy first proposed in the Individualized Instruction for Data Access (IIDA) project should be examined again for this application. [5]

As results are returned, the size of each of the graphic pools is adjusted to be approximately proportional to the number of records in the pools. The overlapped pools representing AND-logic are also kept roughly proportional. The resulting number of hits is posted next to the terms in the pools and to the pools of combined terms.

At any time during the construction and execution of the search logic, the users may interrupt when they see a resulting pool with records they wish to examine further. During the display of records, all keywords contributing to the retrieval of the record are highlighted. The user can stop the scrolling, pick additional terms of interest, using the cursor or the mouse to point and click.

Control of scrolling is handled as a "throttle." Flow control is only simulated, with data continuing to stream into a background task at full speed. Additional terms selected may be added to any of the pools, as directed by the user. The additional Boolean logic is then constructed and issued to the host.

The current state of the search logic in its graphical pool format can be recalled to the foreground at the touch of a key at any time during the display and download session. This allows newly selected terms to be added to the query while records continue to flow in at the front-end. Another key will restore the display of records, starting with the last record seen.

10. Conclusions

Front-ends can be strategically placed on the databank mainframe, a dial-up mainframe, a local mainframe, or a microcomputer. Each placement has its advantages and disadvantages. The microcomputer is best suited for providing the user with a rich interactive environment that can graphically define, control, and present the progress of the search. Combined with AI methods that are becoming increasingly practical for microcomputers, the microcomputer solution to the front-end problem has a very promising future.

REFERENCES

- [1] Hawkins, D.T. and L.R. Levy. "Front End Software for Online Database Searching: Part 1: Definitions, System Features, and Evaluation." Online 9 (May 1985): 30-37.
- [2] Hawkins, D.T. and L.R. Levy. "Front End Software for Online Database Searching: Part 2: The Marketplace." Online 10 (January 1986): 33-40.
- [3] Hawkins, D.T. and L.R. Levy. "Front End Software for Online Database Searching: Part 3: Product Selection Chart and Bibliography." Online 10 (May 1986): 49-58.
- [4] Pemberton, J. "Databank." (Editorial) Online 9 (May 1985): 95.
- [5] Meadow, C.T. and Others. "A Computer Intermediary for Interactive Database Searching. Part I. Design." Journal of the American Society for Information Science 33 (September 1982): 325-332.

AUTHOR AFFILIATION: David Toliver is a Manager, for Software Development and Acting Director, Software Products, at the Institute for Scientific Information (ISI) in Philadelphia. He is also an adjunct Assistant Professor in the College of Information Studies, Drexel University.

ADDRESS: Institute For Scientific Information (ISI)
Software Development, 4th Floor
3501 Market Street
Philadelphia, PA 19104

DEALING WITH HETEROGENEITY:
AN ARCHITECTURE FOR UNIFORM ACCESS/INTEGRATED PRESENTATION

Rita F. Bergman and Diane C.P. Smith

Nowhere is computer-based change coming faster than in the world of the information analyst. Whether that analyst deals with financial data, chemical, petroleum, or military data, whether it be textual, numeric, pictorial, or graphical, the data is or will soon be available on computers. Furthermore, it will be more quickly available, more timely, and therefore more amenable to analytic processing.

However, the forces behind this surge of data capture have created an environment in which the data is separately owned, and remotely stored on a wide variety of storage media in an even wider variety of types and formats. The analyst is faced with the formidable task of mastering a number of access languages (in the worst case, one per data type), resolving or fusing the data when accessed, analyzing it, and integrating the results for reporting and presentation.

The analyst needs tools to handle this heterogeneous environment, and a framework from which to address change as an acceptable constant. Some of the needed tools already exist, developed for special purpose environments - extendible and generalizable. They include heterogeneous database access mechanisms, automatic layout algorithms, and complex data type extensions to conventional database management technology. Furthermore, these tools are appropriate components of any architecture for a full function analyst's information system.

This paper reviews the technologies driving this environment, discusses the underlying problems, and presents a candidate architecture as a solution strategy. Specifically, we discuss the issue of interoperability with respect to remote data sources and complex data types.

Front-End, Interfaces, Gateways, Information Retrieval Systems,
System Architecture

Background

Trends in both hardware and software technology are providing significantly greater capabilities to users of information analysis and decision support systems. Furthermore, the technological trends are inter-related, cause to effect. The introduction of videodisks and C-D ROM provoked the development of retrieval software for those media. Similarly, the increased storage capacity of workstations and PCs has made it possible, as well as desirable, to store and retrieve more complex kinds of data. The increased quality of high resolution graphics has resulted in attempts to display more detailed information.

The accelerated capture of non-conventional data types into electronic form is also a good example of this inter-relationship. As data storage capacity and techniques have expanded, so has grown the desire, and need, to have access to larger volumes of data. Non-conventional data include chemical formulas, molecular structures, full text, graphics, drawings, and images. These data types are considered atypical in that they demand different kinds of operations to be performed on them for purposes of query and retrieval. For example, in the case of maps, the user needs to be able to zoom in for more detail as well as scroll or scan across the topographical surface. In the case of drawings, users must be able to rotate, stretch or otherwise manipulate the object.

The applications affected by the technologies highlighted above range from engineering drawings used in manufacturing and logistics support, to military and intelligence analysis of sensor data, to electronic publishing which encompasses text and photographs.

Two generic sets of problems arise from such diverse, heterogeneous environments. The first is the lack of common or uniform access. Hardware and communications protocols are rarely alike across machines. The lack of standards presents difficulties not just in query but also in storage. Data are incompatible especially across type, and the command languages differ significantly. The second set of problems addresses the difficulties in making an "integrated presentation." The diversity in the kinds of data requires multi-media composition. Photographs, text, as well as numeric data may all need to be combined on the same screen. Aside from handling the different data types is the need to lay out or arrange the physical screen automatically.

Essentially this environment poses a classic tower of Babel syndrome where the diversity and complexity make the development of a coherent system difficult. Not only is it challenging to construct a query across multiple, heterogeneous systems, but it is then an equally difficult problem to determine how to present the results in an integrated fashion.

Technical Approach

Solution strategies for both of these problem areas have been defined and prototyped. The strategies for providing uniform access typically provide solutions within a single complex data type. They are based on the approach of using an extended data model - that can encompass the variations in existing systems.(1) The system concept underlying this approach includes a global view that provides a logical, integrated picture of data from different existing sources. It draws on information located within a global data dictionary, and provides a mechanism for resolving conflicts among data values or representations across systems.

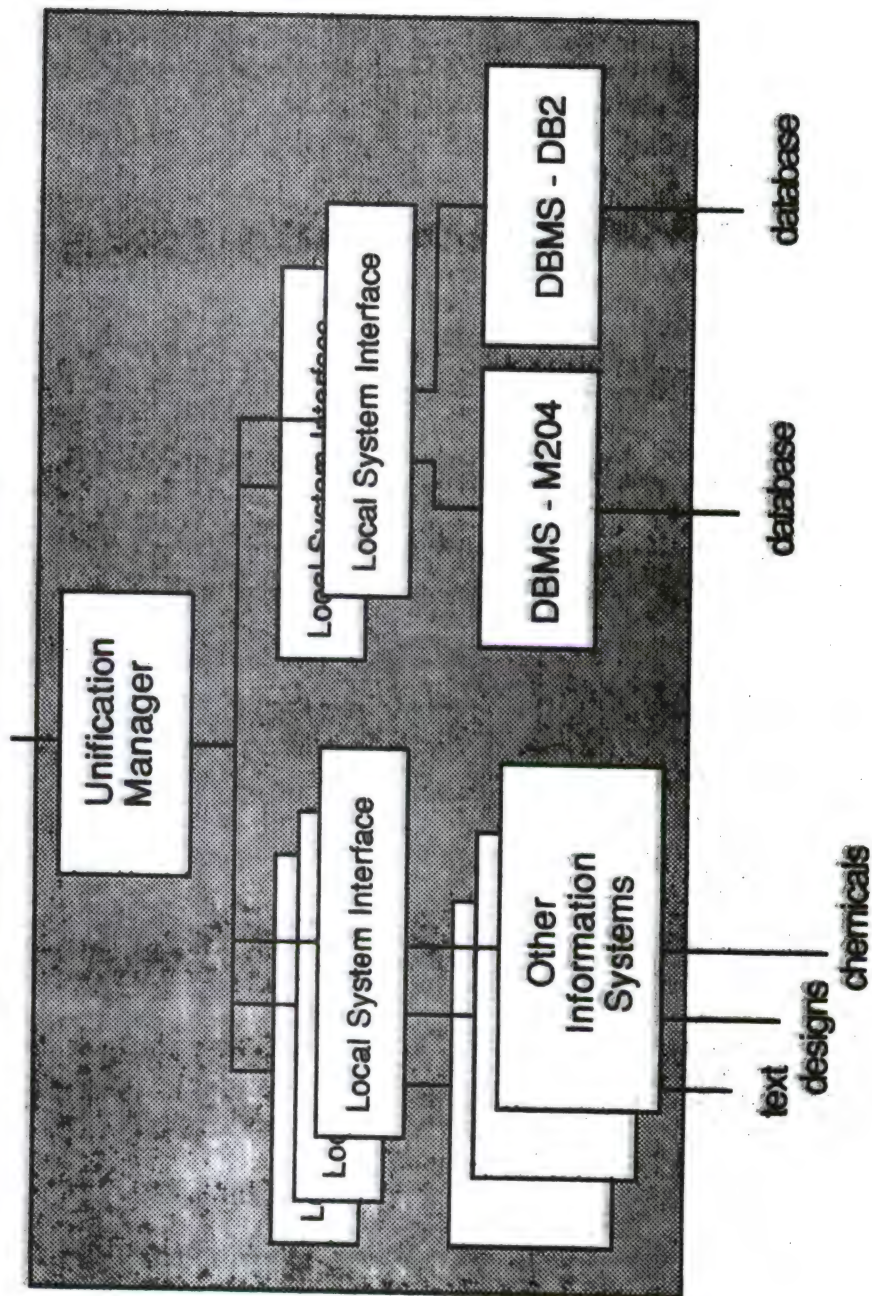
Several operational prototypes have been built using slightly different techniques for handling three kinds of data, e.g., records, represented by commercial DBMSs; chemicals; and designs and drawings for the CAD/CAM environment. MULTIBASE provides integration of different kinds of DBMS records with its use of an extended relational query language. Rules are applied to resolve incompatibilities.(2) CSIN (Chemical Substances Information Network) applied the use of scripts to phrase a query, and applied an internal thesaurus to handle terminology differences.(3) CCDBMS (CAD/CAM DBMS) was designed to support the query of designs and drawings through the use of a hierarchic query language. Incompatibilities were resolved through transforms in which different scales or units of measure were converted to a standard, or in which an algorithm was used to derive a value.

A generic architecture that is applicable to a variety of environments is shown in Figure 1. This architecture provides for a diverse set of interfaces, all of which are handled by a Unification Manager. The Unification Manager fills several key functions, including query handling (drawing on a data dictionary for information about the target systems including what host contains which data), query optimization, and resolving data conflicts or incompatibilities. Local system interfaces receive the parsed query and translate it into sub-queries which can be handled by the target systems. The local system interfaces also receive the sub-results, and transmit them back to the Unification Manager for integration with other sub-results as appropriate.

The second area is in effecting an integrated presentation. The first problem entails handling data from a variety of sources, including maps, sensors, photographs, satellites as well as records stored in a traditional database. Accommodating different data sources is accomplished effectively with the use of high-resolution graphics displays which can present a variety of data types in detail.

GENERIC INTEGRATION ARCHITECTURE

INTERFACES: human, tool, external



The second problem revolves around the large volume of data. Each drawing, map, and photograph, for example, represents several hundred bytes of data. When they are combined, the total volume of data increases rapidly. Three approaches have been applied for handling large volumes of data for presentation to the user. (4) The first is the creation of an organizational framework in which the technique was to create hierarchical information planes or spaces where each successive plane provides another level of detail. Moving across an information plane enabled the user to view what other data exists at the same level. A hierarchical framework is generally an excellent tool for organizing data especially for users who are unfamiliar with the database.

The second strategy entails the use of navigational aids. Even within a hierarchical framework, the sheer volume of information makes it easy to get lost. Consequently, navigational aids are provided. There are three types: passive - where color or sign posts indicate the information level; video - where a second screen display depicts a world-view of the simultaneously with the detailed or local view; and audio - where icons in an information space emit a sound associated with a specific information space.

The third strategy involves layout aids. Layout refers to the ability of a system to create dynamically the appropriate display based on the kinds of information or data types that satisfy a query request. The display must be modifiable by the user, if so desired, and it must be capable of incorporating updated information as well as more detailed information.

The technical solution to developing layout aids relies on knowledge-based graphics and real-time image generation. By this we mean that the graphics display is dependent upon descriptive information available and rules that are resident in the database. This is an area where additional work still needs to be done. (5)

Conclusion

Considerable progress has been achieved in building systems that support integrated access and presentation in a heterogeneous distributed environment. Two remaining technical problems are providing uniform access across data types, and building facilities for integrating the presentation of diverse data. Experimental systems are underway in both areas, particularly in engineering information management, and intelligence applications.

References

- (1) Shipman, David W. The Functional Data Model and the Data Language DAPLEX. ACM Transactions on Database Systems, Vol 6(1): 140-177, March 1981.
- (2) Landers, Terry and Rosenberg, Ronni. An Overview of MULTIBASE. Proceedings 2nd International Symposium on Distributed Databases, H.J. Schneider, ed., Berlin, W. Germany: 1-3 September 1982.
- (3) Bergman, Rita. Beyond SDI: On-Line Retrieval Scripts in the Chemical Substances Information Network. Proceedings of the 44th ASIS Annual Meeting, Vol. 18, pp. 276-278, 1981.
- (4) Herot, Christopher, Carling, R., Friedell, M., Kramlich, D., and Rosenberg, R. Overview of the Spatial Data Management System, November 1981. CCA-81-08.
- (5) Friedell, Mark. Automatic Synthesis of Graphical Object Descriptions. Computer Graphics, Vol. 18(3): 53-62, July 1984.

AUTHOR AFFILIATION: Rita Bergman is a Branch Manager, Research and Systems, for Computer Corporation of America.

Diane C.P. Smith also works for this company.

ADDRESS: Computer Corporation of America
1800 Diagonal Road
Alexandria, VA 22314

End-User Searching in an Online Catalog Environment

William H. Mischo

Many academic libraries, including the University of Illinois at Urbana-Champaign (UIUC) Library, have implemented local online public catalogs to facilitate access to collections. Recent online catalog use studies report that users require enhancements to subject access, including access to periodical literature. While libraries have provided online access to abstracting and indexing services via commercial database vendors, the direct interaction with these information retrieval (IR) systems has been primarily by trained intermediaries. However, academic libraries have begun making available to patrons direct searching of less expensive after-hours versions of the online systems. Studies of these end-user search services show that users are enthusiastic about performing their own searches but experience difficulties with database selection, search strategy formulation, and the use of Boolean operators. In addition, experience with both end-user searching and the optical disk IR systems indicates that users are confused about the need to switch online systems (and to physically move to a different terminal) to retrieve local holdings and availability information.

The UIUC Library presently employs microcomputers with interface software as terminals for the online catalog and circulation system. This paper describes a project to develop interface software to study end-user database searching within an online catalog environment. The microcomputer interface software is designed to: 1) enable untrained end-users to perform database searches as a component of the online catalog and 2) link the retrieved citations to local holdings and availability information within the online catalog. The study will examine the efficiency of online catalogs employing this approach to enhance access to periodicals and explore the cost factors and service requirements associated with implementing this approach.

Microcomputer Interface Software, Online Catalogs, End-User Searching

Two of the most widely hailed and highly publicized developments in both academic and public libraries in the last several years have been: 1) the implementation of online public access catalogs, and 2) the use of online database systems by library patrons or end-users (1-2). This paper will provide a brief overview of present library experiences in these two areas. In particular, these developments will be examined from the viewpoint of the library systems designer and discussed in the framework of two projects implemented at the University of Illinois at Urbana-Champaign Library. These two projects involve the employment of microcomputer interface software to provide: 1) an interface for a linked online catalog/circulation system, and 2) an end-user search system that provides access to remote databases as a component of the online catalog. These systems are designed to augment traditional catalog access, provide expanded subject access, and address the problems encountered with end-user searching of database systems in native mode. The use of microcomputer workstations with interface software to heterogeneous systems addresses the converging trends of online catalogs and online reference (3).

The projects described in this paper involve interfaces to operational command language driven bibliographic information retrieval systems. These systems employ traditional inverted file architecture and vary in the sophistication of their information retrieval capabilities. One of the obvious results of the rapid deployment of online catalogs and end-user based database systems in libraries has been to force library system developers and vendors to design and implement user interfaces and front-ends to these systems. These interfaces have been developed, for the most part, with limited software tools and without the benefit of advanced artificial intelligence techniques. It is interesting to note, however, that online catalogs and library online search services for end-users are the most visible and heavily used computer interfaces for information retrieval systems. For example, the turnkey system vendor CLSI has its bar-coded library cards in the wallets of 25 million people (4).

Figure 1 summarizes the key findings in recently conducted online catalog user studies. Figure 2 focuses on the results of several detailed transaction log analyses. Also of interest is a study utilizing the Northwestern University Online catalog transaction logs which showed that while 37% of the title searches retrieved no results, in 40% of these failed searches the title was in the database (5). Likewise, the study showed that 50% of the unsuccessful author searches were for names and items contained in the database. Clearly, the results of online catalog user surveys and transaction log analyses indicate

difficulties with subject retrieval and, to a lesser extent, with title and author retrieval. The studies show the importance of the interface in retrieval and provide strong evidence that further interface enhancements are necessary. Also of interest is the concern, across all types of libraries, with increasing access to periodical literature.

The University of Illinois at Urbana-Champaign online catalog is comprised of the LCS short-record circulation system and the WLN search software operating on full MARC records loaded from OCLC archive tapes. The locally developed modules include a linking file and a public interface running on IBM PC's. The interface software employs some navigated searching techniques to facilitate subject retrieval. For example, Figure 3 shows the initial subject authority display for a search for the subject COMPUTER VISION. From the displayed list of alphabetically closest subject headings and menu options, the searcher can press ENTER to try another search when the displayed headings are not satisfactory. This prompts the interface to perform a title word search as illustrated in Figure 4. In this example, 21 records were retrieved with the title keyword search capability. This example illustrates the problems of left hand title string matching and the value of title word searching as a feature in online catalogs. Selection by the searcher of a full record, as shown in Figure 5, allows further subject retrieval when the searcher chooses to find relevant headings. The interface then extracts the first LCSH from the displayed record and performs an authority file search on the extracted heading for display, as in Figure 6. In this way, the searcher can expand the search by utilizing the (broader) subject heading to retrieve related items which may not contain the title words COMPUTER VISION.

Because the Illinois online catalog employs microcomputers as public terminals, the capability is present to access the local online catalog, other off-site online catalogs, remote dial-access bibliographic databases, and information stored on microcomputer files (Figure 7). The initial menu for the locally developed front-end ILLINOIS SEARCH AID is shown in Figure 8. This software serves as a tool for public service librarians and operates on PC's with 2 communication boards--one for direct access to the online catalog, the other for dial access to remote databases. The software search features include: offline entering, editing, and storing of search strategies; (with some error checking of command strings); uploading and execution of offline strategy; automatic logon with making of passwords; selective downloading and printing (function key controlled); typing ahead; and offline printing with post-processing. ILLINOIS SEARCH AID also provides the capability of storing and displaying reference information databases and a simultaneous link with both a database vendor and the online catalog via function key control.

Libraries have provided online access to commercial database vendors since the late 1960's, but until recently the direct interaction with these systems has been almost exclusively by trained intermediaries. However, libraries have begun making available to patrons direct searching of the less expensive after-hours versions of the online systems. Studies of these end-user search services (summarized in Figure 9) show that patrons are enthusiastic about performing their own searches, but experience difficulties with database selection, search strategy formulation, and the use of Boolean operations. In addition, experience with both end-user searching and the optical disk IR systems indicate that users are concerned about the need to switch online systems and to physically move to a different terminal to retrieve local holdings and availability information.

The remainder of this paper will describe the work to date on a project to provide end-user searching in an online catalog environment. This project, funded by IBM and CLR, will provide interface enhancements to allow untrained users to perform database searches as a component of the online catalog and link retrieved citations with local holdings and availability information within the online catalog. Methods of linking citations dynamically under software control to online catalog records are being explored, and the interface will also allow users, after performing database searches, to "switch" over to accessing the online catalog to retrieve call numbers and holdings.

Library options for providing direct user access to the periodical literature are listed in Figure 11. These include: loading databases into online catalogs, CD-ROM, and end-user searching of vendor systems. This paper deals with the third option; however, all of these options are being explored at the University of Illinois Library.

Figure 12 shows the initial menu for the enhanced online catalog interface. Users choosing to retrieve citations within periodicals are presented the screens of information on online searching shown in Figures 13 to 15. The user is asked to supply search concepts as illustrated in Figures 16 to 19. This interface is designed to allow simple, straightforward searches, not to provide comprehensive retrieval. Its goals are to address the identified problems end-users experience with search strategy formulation and Boolean operators and provide more effective searches within constraints of online costs and staffing costs. The interface uses offline storage of search strategy and software controlled search navigation which performs sets combination, selection of optimum result set, downloading of selected citations, and automatic logoff from the database vendor with subsequent post-processing and printing of references. This process is shown briefly in Figures 20 and 21. In this way,

online time and costs are strictly controlled. The software strategy employed for searching the ERIC database is shown in Figure 22. This heuristic model for searching this database illustrates the problems of navigated searching in sophisticated information retrieval systems employing proximity operators, field delimiters and free text and controlled vocabularies. These specific database-dependent strategies are a limited application of the typology suggested by Harter and Peters (6).

Future plans for the interface include the development of an online feedback mechanism to allow user intervention and modification in the search process. A detailed study of the interface software will examine the efficacy of online catalogs employing this approach to enhanced access to periodicals. Also explored will be the retrieval effectiveness, cost factors, and service requirements associated with this approach to end-user searching.

CLR sponsored Online Catalog Public Access Project.

Insights concerning subject access gleaned from user questionnaires, transaction log analysis, and focused group interviews.

- 1) Subject searching is the predominant mode of searching; in some libraries it accounts for more than one-half of all searches.**
- 2) The majority of catalog users want materials on a topic.**
- 3) Catalog users report more problems with subject searching.**
- 4) Users approach online catalogs with the expectation that they will find enhanced access to a broader field of materials than in the card catalog.**
- 5) Catalog users place the highest priority for improvements on various subject search enhancements.**
- 6) One third to one-half of searches result in nothing retrieved.**
- 7) Subject searches using keywords with a non-Boolean strategy or with search arguments providing a partial match with controlled vocabulary terms often produce a large number of citations.**
- 8) Online catalog users exhibit more perseverance than traditional card catalog users.**

Further analysis of the Online Catalog Project data by Mathews and Lawrence.

- 1) Results suggest that a different system interface may be needed for subject searching.**
- 2) Catalogs should display an intermediate index of subject headings in response to a subject search argument.**
- 3) Systems with keyword searching receive more subject searching.**

Figure 1

Transaction Log data

Analysis of Syracuse University Online Catalog transaction logs (Markey).

- 29% of searchers' input matched or were close matches of LCSH.
- In 65% of searches, there was at least one access point that resulted in no retrievals.
- Of the LCSH matches 29% retrieved 100 or more citations.
- 36% of the access points were "whatever popped into the searcher's mind"; this occurred in 61% of the searches.

Analysis of Dewey Decimal Classification Online Project alphabetical searches.

- 28% of patron and staff search terms matched LCSH.
- 2% of patron and staff search terms matched Dewey Relative Index entries.
- Hyphenation, punctuation and abbreviations were a problem.

University of Illinois transaction log analysis (preliminary).

- Less than 45% of user entered search terms were even partial matches with LCSH.
- over 62% of subject searches were forced into the title word search option.
- A large percentage of the LCSH partial matches did not yield a useful intermediate subject index display.

Previous catalog use studies.

- User initial entry vocabulary matches subject catalog c. 50% of the time.
Note problems of studies in dictionary catalogs and partial matches.
- Phenomenon demonstrated by Batos concerning term matches with more general subject headings in which users fail to locate specific relevant heading.
"Mental tests" instead of "Rorschach Test".

Figure 2

(Press <ENTER> if you aren't sure of the wording.)

05:04:15

Press <ENTER> for all or press the corresponding key if the subject is about a topic--<T>, person--<P>, corporate--<C>, or geographical area--<G>.

Searching the Full Bibliographic Records of the holdings acquired since 1975.
AUTHORITY DISPLAY

1. Computergraphics.
- * 2. Computerized Library Advanced Package
3. Computerized mapping.
- * 4. Computerized reservation systems
- * 5. Computerized Resources Information Bank
- * 6. Computerized serials systems
7. Computerized type-setting--Style manuals.
- + 8. Computerized typesetting
9. --Congresses.
10. --Handbooks, manuals, etc.

These are the closest subject headings. Do one of the followings:

Type a number and <ENTER> to see the corresponding bibliographic records.

Press to browse--to see more headings.

Press <E> to end this search.

Press <I> for an interpretation of the symbols to the left of the headings.

If these headings are not satisfactory, press <ENTER> to try another search.

Subject Search: COMPUTER VISION

Figure 3

No items were found.

Trying to find COMPUTER VISION in titles...

Found 21 records.

BIBLIOGRAPHIC DISPLAY

1. Fundamentals in computer vision : an advanced course / 1983. x, 498 p. ocm08-709313
2. Structured computer vision : machine perception through hierarchical computation structures / 1980. xi, 234 p. ocm06-277740
- 5 3. Computer vision, graphics, and image processing. Vol. 21, no. 1 (Jan. 1983) 1983 v. ocm08-690623
4. An overview of computer vision / Gevarter, William B. 1982: ocm09-462171
5. Proceedings of the Workshop on Computer Vision--Representation and Control, August 23-25, 1982, . . . Workshop on Computer Vision: Representation and Control c1982. vi, 185 p. ocm09-083081

These are short records number 1 - 5. Do one of the following:

Press <C> for circulation information.

Type a number and <ENTER> to see the corresponding full record.

Press <E> to end record display.

Press <ENTER> without a number to see the remaining records.

Subject Search: COMPUTER VISION

Figure 4

hierarchical computation structures / 1980. xi, 234 p.
ocm06-277740

05:06:31

- 5 3. Computer vision, graphics, and image processing. Vol. 21, no. 1
(Jan. 1983) 1983 v. ocm08-690623
4. An overview of computer vision / Gevarter, William B. 1982:
ocm09-462171
5. Proceedings of the Workshop on Computer Vision--Representation
and Control, August 23-25, 1982, . . . Workshop on Computer
Vision: Representation and Control c1982. vi, 185 p.
ocm09-063081

BIBLIOGRAPHIC DISPLAY

Fundamentals in computer vision : an advanced course / edited by
O.D. Faugeras. Cambridge :Cambridgeshire: ; New York : Cambridge
University Press, 1983.

x, 498 p. : ill. : 24 cm.

Includes bibliographical references. ISBN 0521250994

1. Image processing--Digital techniques--Addresses, essays,
lectures. I. Faugeras, O. D.

TA1632 .F84 1984

621.36/719

DLC ocm08-709313

Press <H> to make another attempt to find relevant headings.
Or press <ENTER> to go on.

Subject Search: COMPUTER VISION

*illustrates
problem
of
specificity
in
LCST*

Figure 5

1. Image processing--Digital techniques--Addresses, essays,
lectures. I. Faugeras, O. D.

05:07:02

TA1632 .F86 1984 621.36/719

DLC ocm08-709313

A relevant heading is: Image processing
Found 13 headings.

AUTHORITY DISPLAY

- + 1. Image processing
 - 5. --Addresses, essays, lectures.
 - 7. --Congresses.
 - 4. --Congresses.
 - 8. --Data processing.
 - 9. --Congresses.
 - 2. --Digital techniques
 - 3. --Addresses, essays, lectures.
 - 6. --Congresses.
 - 10. --Equipment and supplies--Congresses.
- These are headings number 1 - 10. Do one of the following:
Type a number and <ENTER> to see the corresponding bibliographic records.
Press to browse--to see more headings.
Press <E> to end this search.
Press <I> for an interpretation of the symbols to the left of the headings.
Press <ENTER> without a number to see the remaining headings.

Subject Search: COMPUTER VISION

Figure 6

TERM PAPER CLINIC ANALYSIS

Analysis of patron request and concerns:

- 1) Users want current information.**
- 2) Users prefer journal articles and recent books.**
- 3) Users want evaluative information, including overviews and critical reviews.**

Microcomputers as intelligent workstations in distributed information network.

Used in public service setting to access:

- 1) local online catalog (hard-wired).**
- 2) other online catalogs off-site (gateway link or dial-up).**
- 3) remote bibliographic databases (dial-up).**
- 4) local databases (gateway link or dial-up).**
- 5) information stored in microcomputer disk files.**

Figure 7

ILLINOIS SEARCH AID

Database Vendors or Networks:

0. Online Catalog--Direct Connect
1. BRS--Telenet
2. BRS--Tymnet
3. Dialog--Telenet
4. Dialog--Tymnet
5. OCLC--Tymnet
6. OCLC--Telenet
7. RLIN--Tymnet
8. SDC--Telenet
9. SDC--Tymnet
11. Resume Searching--Already Online
12. Print previously downloaded data
13. Simulation of online search
14. Logon by Searcher
15. Online catalog--Dial-up
- **16. Reference Information
17. Knowledge Index--Telenet
18. Knowledge Index--Tymnet

CHOOSE ONE OF THE NUMBERS 16

Figure 8

End User searching facilities in libraries.

Studies of end user search services show:

- 1) Very popular service.**
- 2) Very staff intensive; require additional (reference) staff to assist in logon procedures, database selection, and search strategy formulation.**
- 3) Users have difficulty with simpler interfaces provided for after-hours services such as BRS After-Dark and Dialog's Knowledge Index.**
- 4) Users have difficulty with Boolean logic & search strategy formulation.**
- 5) Searches performed with intermediaries still yield better results.**

Economic issues and trends (Southern Cal Univ, Texas A & M, Univ of Ottawa, Dartmouth University).

Role of Library in end user searching.

Menlo's In-Search problems and future of end user interface market.

University of Illinois project (supported by IBM and CLR) on end user searching in online catalog environment.

- 1) To enable end users to perform database searches as a component of the online catalog.**
- 2) To link retrieved citations with local holdings and availability information.**
- 3) To utilize microcomputer workstations to provide expanded subject access in an "analytic" online catalog.**

Figure 9

Online database searching in libraries.

The number of publicly available databases has grown from 301 in 1976 to 3,010 in 1985.

The number of online records has grown from 52 million in 1976 to 1.7 billion in 1985.

The major library vendors provide access to over 200 million bibliographic citations.

In 1982, library and information center connect time usage totalled over 1.5 million hours.

Figure 10

Library public service options for direct user access to the periodical literature.

1. Load databases into local online catalogs.
Limitations: number of databases and records; lack of sophisticated retrieval software; costs; duplication.
2. CD-ROM or other optical storage technology.
Limitations: Number of databases and records; lack of sophisticated retrieval software; costs.
3. Provide gateways to database vendors.
Limitations: costs; telecommunication factors.

Figure 11

UNIVERSITY OF ILLINOIS ENHANCED ONLINE CATALOG SEARCH OPTIONS:

1. Search Online Catalog for titles in Library collections; look for books, journals, reports, theses, etc. by author, author/title, subject, call number).
2. Retrieve citations to articles within journals and magazines, that is, search a periodicals index like Reader's Guide online in order to retrieve citations to articles on a topic.

TYPE "1" OR "2" TO CHOOSE

Figure 12

PLEASE READ THE FOLLOWING INFORMATION CAREFULLY.

This database search of the Education database will retrieve citations to articles in periodicals (journals and magazines), papers read at conferences, and government documents.

Below is a sample of a retrieved citation from the May 1984 ONLINE magazine:

AU (Author) Hunter, Janne A.

TI (Title) When your patrons want to search--the Library as advisor to end users. A compendium of advice and tips.

SO (Source) Online; v8 n3 p36-41 May 1984. 84.

YR (year) 84.

DE (Descriptors) Information-Retrieval. Library-Services. Online-Systems. Training-Methods. Computer-Software. Reference-Services. Search-Strategies.

AB (Abstract) Most citations include a short abstract or summary.

PRESS ANY KEY TO CONTINUE.

Figure 13

The most important aspect of a database search is search strategy formulation. The search strategy consists of a series of phrases or keywords entered by the searcher to describe the topic.

The online retrieval system used here searches the database of article citations to retrieve citations containing the searcher's keywords. These keywords can appear in the article title, subject descriptors, or abstract.

TYPICAL SEARCH TOPICS:

1. Phase transformations in stainless steel in a hydrogen environment.
2. CAD/CAM using microcomputers.

Notice that the search strategy words can be grouped into different concepts or components. In the first example above the 3 concept classes are: 1) phase transformations; 2) stainless steel; and 3) hydrogen. In the second example, there are 2 concept groups: 1) CAD/CAM; and 2) microcomputers. A typical search will have from one to three concepts.

PRESS ANY KEY TO CONTINUE.

Figure 14

In the example search:

"Phase transformations in stainless steel in a hydrogen environment"
there are 3 concept classes:

- 1) Phase transformations;
- 2) Stainless steel;
- 3) Hydrogen.

Other words or phrases may be synonyms or alternative phrases for each of the concepts. For example, for "phase transformations", the terms "phase diagrams" or "phase shifts" could be substituted as synonyms. For the concept "stainless steel", the searcher might also be interested in articles with the additional terms "titanium" or "304 steels".

The search strategy then could be represented as:

CONCEPT 1	AND	CONCEPT 2	AND	CONCEPT 3
Phase Transformations		Stainless Steels		Hydrogen
OR		OR		
Phase Diagrams		Titanium		
OR		OR		
Phase Shifts		304 Steels		

PRESS ANY KEY TO CONTINUE

Figure 15

We will need to identify the concept terms for your search request. The terms in these concept groups will then be used to find references to journal articles and reports that are relevant to your request.

Below type what you consider the ideal title for an article that would best meet your information needs.

(Type title followed by ENTER key):

end user searching in libraries

Using these title words as a guide, you need to identify the major concepts making up your search topic.

Below type a word or phrase describing concept 1 of your search request. Do not include articles ('a', 'an', 'the') or prepositions (of, on, by). Use single words or phrases such as 'stainless steel' or 'dual phase steel'. Type the word or phrase followed by the ENTER key.

Concept 1 should be the central or most important topic of your search.
Concept 1: end users

Figure 16

Type below any synonyms (dual phase steel, two phase steel), alternate spellings (sulphur, sulfur), or additional related or alternative terms (titanium, aluminum, niobium) for concept 1: 'end users'.

Be particularly aware of adding specific types within broad concepts, e.g. for the concept 'heavy metals' add specific types such as 'lead', 'cadmium', etc.

Type the additional concept terms one at a time.
If there are none or no more, just press the ENTER key alone.
Additional term (word or phrase): direct users

Examine the entered word/phrase: 'direct users'
for spelling errors or typos.
If search term is correct press ENTER key alone. If you wish to retype the term do so below.

Figure 17

For your title:
'end user searching in libraries'
The terms making up concept 1 are:
end users or direct users

This will be searched in the language of the database system as:
end with user\$2 or direc\$2 with user\$2

Below type a word or phrase describing concept 2 of your search request.
Do not include articles ('a', 'an', 'the') or prepositions (of, on, by).
Use single words or phrases such as 'stainless steel' or 'dual phase steel'.
Type the word or phrase followed by the ENTER key.

Press the ENTER key alone when you are finished entering search concepts.
Concept 2:

Figure 18

The search strategy is listed below.
end users or direct users
AND
libraries
AND
microcomputers

Please examine these search terms.
If there is a problem and you wish to stop the search
and begin entering the strategy again type 'STOP' and press ENTER.
Else press only ENTER key to go on.

Figure 19

Logon completed.

You will now be connected to the Education database which contains citations to journal articles, documents, conference papers, and books from 1966 to the present. Each concept group will first be entered separately, then combined in different ways in order to yield the best search results.

Now searching search concept 1:
end users or direct users

The terms in concept 1 appear in 252 documents.

Now searching search concept 2:
libraries

The terms in concept 2 appear in 25882 documents.

The retrieved sets of references from each concept group will be compared to determine which references have concepts in common.

Now searching for articles that have at least one term from each concept appearing in the title, descriptors, or abstract.

Recv

N

Figure 20

Now searching for articles that have at least one term from each concept appearing in the title, descriptors, or abstract.

There are 112 references that contain terms from all concepts.

Now searching for references with terms from each concept appearing in the title or in the same sentence in the descriptors or abstract.

There are 74 references that contain terms from all concepts in the same sentence.

Now searching for references with terms from each concept appearing only in the title or descriptors.

There are 32 references with terms from each concept appearing in the title or descriptors.
The most recent 20 references will be first transferred to floppy disk and then printed.

Now printing to disk 20 references.
This will take about 45 seconds.

Recv

N

Figure 21 259

SEARCH STRATEGY FOR ERIC DATABASE ON BRS AND BRS AFTER-DARK.

For concepts C1, C2, ... Cn
where Ci=T1 with T2 or T3 with T4 or T5 with T6 with ...
for a maximum of three concepts.

Navigated search strategy:

I. Postings for each concept are obtained.

IF Cn=0 for n > 1 then Cn is eliminated.

IF C1 < 35 then print most recent 25 citations; End search.

II. Concepts are combined with 'and' operator free text, i.e., C1 and C2 and ... Cn.

IF result > 40 then proceed to step III.

IF result = 0 then make combinations C1 and C2; C1 and C3; ... C1 and Cn (presently maximum n is 3).
Use the largest set for further processing.
IF all combinations are still 0 then End search.

FOR result <= 40 then print most recent 25 (download to disk, then print with local info after logging off. End search.

III. Concepts are combined using 'and' operator limiting concepts to .ti,de,id. C1.ti,de,id. and C2.ti,de,id. and ... Cn.ti,de,id.

IF result > 40 then proceed to step IV.

IF result = 0 then print most recent 25 of step II.

FOR result <= 40 then print most recent 25. End search.

IV. Concepts are combined using 'with' operator free text, i.e., C1 with C2 with ... Cn. IF n=1 (only one concept) then this step is not executed; rather most recent 25 from step III are printed.

IF result > 40 then print most recent 25 from this set IV or set III, whichever is smaller.

IF result < 15 then all printed; this set is then 'not'ed (subtracted) from set in step III and most recent 15 are printed; End search.

ELSE print most recent 25; End search.

Alternatives to IV are limit 'with' to ti,de,id; 'or' with III.

REFERENCES

1. Mathews, Joseph R. et al., eds. Using Online Catalogs: A Nationwide Survey. New York: Neal Schuman; 1983.
2. Janke, Richard V. "Online after Six: End-User Searching Comes of Age," Online 8:15-29 (November 1984).
3. Butler, Brett. "Online Catalogs, Online Reference: An Overview," in Aveney, Brian and Butler, Brett, eds. Online Catalogs, Online Reference: Converging Trends. Chicago: American Library Association, 1984, pg. 1-19.
4. Miller, Tim. "SilverPlatter: Dishing up Data for Libraries," Information Today 3(6):24 (June 1986).
5. Dickson, Jean. "An analysis of User Errors in Searching an Online Catalog," Cataloging and Classification Quarterly 4(3):19-33 (Spring 1984).
6. Harter, Stephen and Peters, Anne Rogers. "Heuristics for Online Information Retrieval: a Typology and Preliminary Listing," Online Review 9(5):407-424 (1985).

AUTHOR AFFILIATION: Bill Mischo is Engineering Librarian at the Engineering Library, University of Illinois at Urbana-Champaign.

ADDRESS: Engineering Library, University of Illinois
222 Engineering Hall
1308 W. Green Street
Urbana, IL 61801

THE IMPACT OF NETWORKING STANDARDS ON LIBRARY SYSTEMS

Michael Monahan

Within the past year several major library standards have either been implemented or reached the final draft form. This paper will discuss these standards and their impact on the operational world of libraries. It will be argued that the library world has within its grasp the possibility of virtually unlimited interconnections. The technology and product support appear to be solid. There is strong and widespread support within the library turnkey vendor community for at least some networking standards. To a large extent the remaining issues are organizational, especially as it will be noted that multipoint peer to peer networking is conceptually different from the centralized, system specific, networks that are currently operational.

Networking, Standards, Libraries

Background

By now almost everyone is familiar with the ISO/OSI (International Standard Organisation / Open System Interconnect) model for computer networking. For context, let us just note that this model is an international standard that allows different types of computer to communicate in a networking, equal to equal, basis. The standard also introduces the crucial concept of layering, so that details of physical interconnection are held separate from application specific requirements.

The ISO/OSI standards have achieved widespread acceptance, as can be seen in the implementation of X.25 packet switching networks. But the use of the lower levels of the ISO/OSI model does not immediately answer all networking issues. Within the ISO/OSI model, the highest layer is retained for application specific definitions. Until appropriate higher level standards are created, ISO/OSI is merely a framework within which general discussions can take place.

A specific attempt to supply higher level application layers of interest to libraries is the activities of the Linked System Project (LSP). In the LSP program, four of the largest bibliographic entities in the United States have defined application layer protocols for library purposes. The four LSP participants are the Library of Congress, the Research Library Group (RLG), the Western Library Network (WLN), and OCLC. The initial use of the LSP implementation was to transfer authority records. Clearly this implementation opens doors beyond merely transferring authority records between the original participants. (1), (2)

The Present

The beginning of operational activity within the four LSP participants has lead to a burst of recent activity in the networking area. Last year the technical committee of LSP (LSP/TC) was expanded to include five organisations with an interest in LSP implementations. The new members are: Geac, Northwestern University, National Library of Medicine, the National Library of Canada, and the Triangle Research Library Network. All of the new members of the LSP/TC are actively working toward a LSP implementation, although they will not necessarily be involved in the authority transfers of the first four participants.

The first announced result of this extension occurred in January of this year, when Geac and RLG announced a phased

program that will allow RLG members who are also Geac sites to transfer records into and from the RLG system using the LSP protocols. At the same time, it was announced that Geac had already transferred records from the RLG computer in California to a system at Geac's head office in Markham, Ontario. This transfer was interesting because it appears to have been the first use of LSP outside the original four participants. It also involved a linkage between the Canadian X.25 and the American X.25 networks. Finally it involved the transfer of bibliographic, not just authority records. (3)

In April of this year, Geac installed a new system at New York University in New York. New York University is a member of RLG and one of the functions of the new system at New York University will be the first production use of the LSP transfer from the RLG computer. Experimental transfers have begun and production activity is expected to occur shortly.

Following on the heels of the expansion of the Linked Systems Project / Technical Committee, was the formation in January of this year of the Automation Vendor Interface Advisory Committee, AVIAC. (4) The AVIAC group is an ad hoc committee that represents the major North American library automation vendors, along with external user representation. AVIAC has as an avowed aim the encouragement of standard development and implementation among the library community. All of the member firms in the AVIAC group are committed to implement the standards that are adopted.

Members in the AVIAC group are CLSI, DRA, Geac, INLEX, and VTLS. Membership is limited on the theory that only small committees actually achieve anything. In fact, most of the other firms active in the North American library automation industry indicated, at the time AVIAC was formed, willingness to join the group. This willingness among potential members implied implementing the standards agreed. The firms listed above were selected using a more or less random method from the ones that were interested in joining. Between the Linked System Project Technical Committee and the AVIAC group, the member agencies represented more than 85% of the medium to large local systems installed by North American vendors in 1985. (5)

LSP has been primarily involved with the transfer of records. At the same time the National Library of Canada has been developing a draft standard within the ISO/OSI framework for interlibrary loan. This draft has been implemented in a test mode between the National Library of Canada and a number of libraries throughout Canada. (6) It is expected that the resulting protocol will be submitted as a standard. There has

been a de facto division of labour with the Americans working on the record and authority transfer areas and the Canadians working on the interlibrary loan specifications. These efforts appear to be complementary, with LSP participants being updated on the interlibrary loan efforts of the National Library of Canada and the National Library of Canada being on the Linked System Project / Technical Committee.

It thus seems reasonable to argue that the climate is right for the North American library automation community to adopt ISO/OSI standards in general and LSP in particular. Of course the translation of this public commitment into operational software is another matter.

At the same time that ISO/OSI standards are being developed there have been a number of experimental systems that address intersystem networking through intermediary systems.

The IRVING project in Colorado links dissimilar systems using "nodal" processors. IRVING uses the lower levels only of the ISO/OSI protocols and faces a severe limitation because of the need that the network be invisible to the library application systems. IRVING has the distinction of being the first such implementation in the library community and deserves credit for expanding the awareness of the possibilities of local system linkage among librarians. The IRVING implementation implies "nodal" processors in addition to the local library system. Because many of the local library systems are now developing comparable levels of networking sophistication it would appear that the precise IRVING implementation will not see general use. (7)

A more recent experimental implementation has occurred at the Defense Technical Information Center (DTIC). In this implementation, a local library system was married to "gateway" software to allow the library system to operate on multiple information sources. This has the significant cost advantage that the library and gateway software share the same computer. The resulting system becomes a significant resource addressing both internal and external databases. (8)

Variations on the DTIC experiment have been conducted by several of the vendors of local library systems. Typically these experiments allow terminals connected to the library computer to become devices on the external network. The level of protocol conversion and the intelligence of the interconnect vary.

The (Near) Future

One of the major benefits from the use of the ISO/OSI model is that it makes possible a "network of equals". By this is meant that the linkages between institutions are not predefined by the network nor is there any bias in the network toward a central control.

The "network of equals" comes from the fact that the actual carrier of messages is the public packet switching network. Here all pay on a per transaction basis and all have equal access to the network. Anyone can implement the public protocol and hence be eligible to participate in the network. The resulting computer network is broadly speaking compatible with the normal view of library information. It means that the broadest possible number of users and libraries have the ability to access the largest possible number of other information sources.

Because the network allows essentially any interconnect, it becomes possible for different national and regional groups to define their own preferred or default interconnects. To use a Canadian example, the 15 university libraries in Ontario might form regional groups based on their location, with a link to the other Ontario universities, followed by a link to the National Library of Canada. This could be the general model, the two university libraries actually in Ottawa might link to the National Library before linking to the other universities.

The presence of such a link can be used for resource sharing of bibliographic records, for inter library loan requests, for acquisition pre order searching (why buy an expensive item if the neighbouring library already has it?), for searching by library patrons. Because the ISO/OSI model is an international standard it becomes possible for more institutions to adopt it. Like a snowball rolling down a hill, the more institutions that make it available the more application options that open and the higher the benefit to each institution.

As the computer to computer networks grow there will be increased pressure on libraries to reach consensus on the meaning of access points. The existing standards largely leave this question open and most local systems can be expected to solve the problem by allowing the library to select the data elements that are included within each index. One implication is that searches using the same index and the same terms may retrieve different results between libraries, even when the material contained in the libraries is the same. As valuable as the standards for searching are, we do not do our clients a service if we fail to admit and understand the

implications of using the same name to describe different searches.

The existence within the library world of such public and wide spread standards will also begin to change the role of the library within its institution. The same standards that allow the library to talk to the external world will allow different systems within the institution to talk to the library.

If a faculty member at an academic institution wants to search the library database and retrieve records for use within his or her personal files, then there will be a publically available standard that the workstation or personal computer can use. Frankly there will be some resistance to using the ISO/OSI model for this purpose. Many current generation micro computers will find the networking model used in the ISO/OSI model a trifle complex and there will be pressures that a "simpler" model be developed. The problem is that such a simpler model will always fall victim to its lack of generality. There will be software for some micros, not others, one will be able to link to brand "X" at the local library but not brand "Y" at the neighbouring library.

There will also be a subtle shift in the function of the library as more and more systems adopt the ISO/OSI model. The library will become an intermediary in the computer information networks that many users access. The library will be able to offer a single access node for the local users that will then pass the users through to other sources of information. In some cases the user will not even be aware of the databases that are searched by the library computer to satisfy the user's request. Just as now many users are unaware of the reference sources consulted before the reference librarian provides a specific reference.

The overlapping information sources that libraries access will mean that no library specific standard will ever supply all of the linkages that a library or its users may want. This does not mean that such an ISO/OSI conforming standard to address library information needs is not valuable. As more of the information sources can support such standards the easier it will be to manage the few that do not.

There exist other issues that we must come to grips with. Unfortunately, record ownership is one of them. When a terminal searches a database it is possible to distinguish between displaying a record and the functions that involve copying the record. In a computer to computer network any record that is transferred for display may be retained and operated upon. In effect there is no distinction between

looking at a record and using it.

Commercial database systems are already used to such issues. But the bibliographic utilities that are part of the LSP all run dedicated, limited access, networks. The switch from such a centrally controlled environment to a "network of equals" will have significant impacts. The development of public positions on record usage is expected. The library community must act to ensure that these position balance the maximum open usage and interchange of data with the legitimate need of networks for fiscal solvency.

Another issue that must be faced is that libraries may wish to use multiple sources of cataloguing material. In the hypothetical Ontario university network used above, the default search paths might also be the default cataloguing source paths. Even more likely is that there might be differences based on the material at hand: music record are searched at library "x" but maps at library "y". The variations in usage of bibliographic records that found among different types of libraries is something that large, nation spanning, utilities will always have great difficulties supporting. The larger and more diverse the libraries using the utility, the greater the problem of support. By contrast, special interest networks based on subject, geography, or simply similarities of practices become entirely possible within the ISO/OSI model.

The practical problem is to convince the utilities to assist with such a development. The utilities must not attempt the impossible of being all things to all people. At the same time there must be developed funding structures that will provide support for any necessary central activities. It is to be hoped that structures created for the specific purpose of prompting resource sharing will not become major obstacles in the effective realisation of the options made available by new technology.

Conclusion

National X.25 networks can already talk to each other. If libraries follow on and implement the ISO/OSI model and the LSP protocols then the members of the international library community will be able to talk to each other. All of us will benefit from this interconnection and therefore all of us have an interest in taking the steps that are possible to us to achieve it. The nature of the standards proposed means that local, regional, and national interests and priorities can be maintained within the resulting environment.

Footnotes:

(1) Ray Denenberg, "Linked Systems Project, Part 2: Standard Network Interconnection", Library Hi Tech, (Issue 9, 1985) pp 71 - 79

(2) Henriette D. Avram, "The Linked System Project: Its Implications for Resource Sharing", Library Resources & Technical Services, (January / March 1986), pp 36 - 46

(3) Advanced Technology / Libraries, Knowledge Industry Publications, (May 1986), pg 1

(4) Library Systems Newsletter, Library Technology Reports, (March 1986), pg 1

(5) Joseph R. Matthews, "The 1985 Automated Library Marketplace", Library Journal, (April 1986), pp 25 - 35

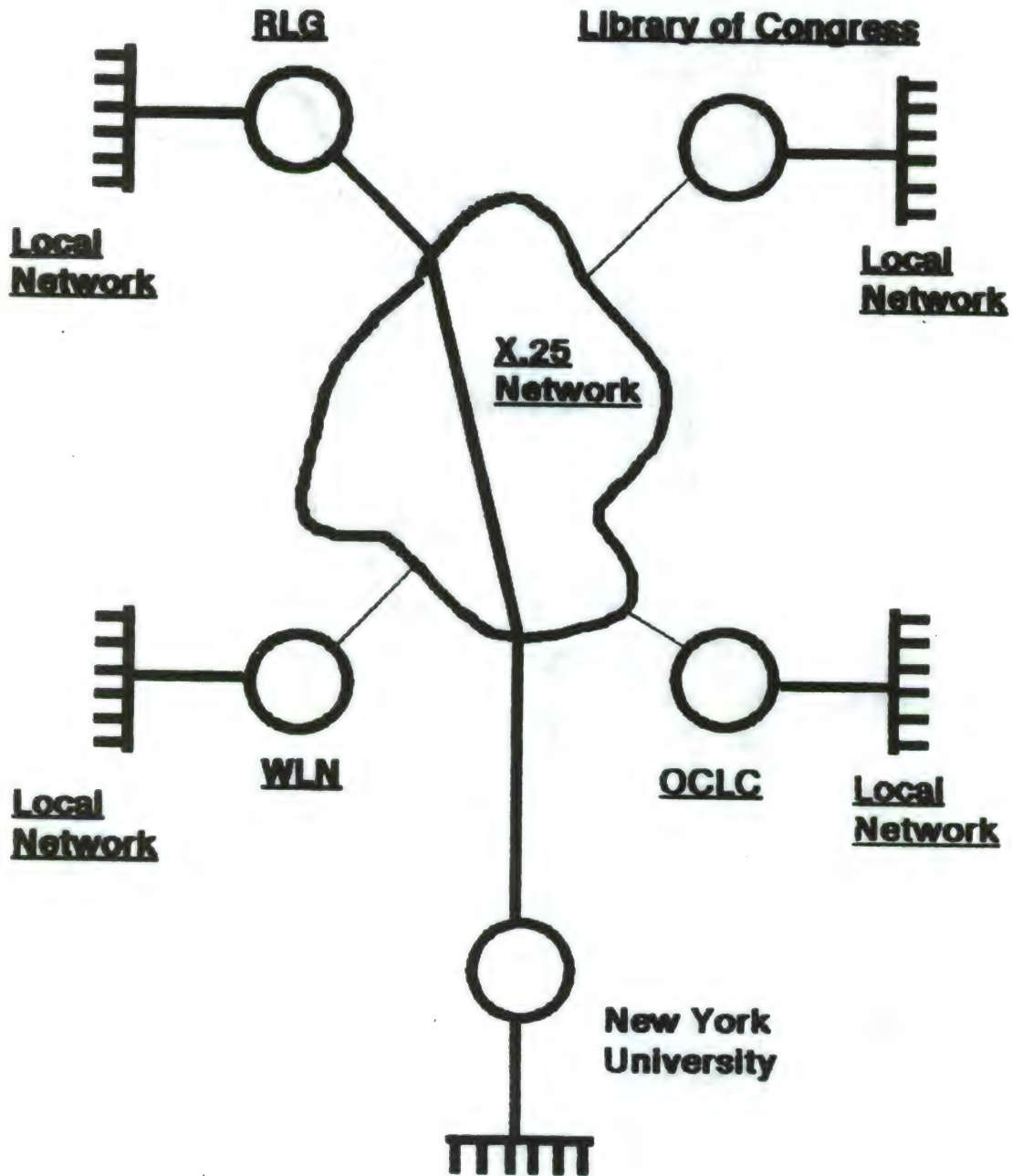
The percentage was taken by including NOTIS and the turnkey vendor numbers together. Systems with less than 8 terminals were excluded.

(6) Office for Network Development, National Library of Canada, "Interlibrary Loans (ILL) Service Definition: Draft", 21 January 1986

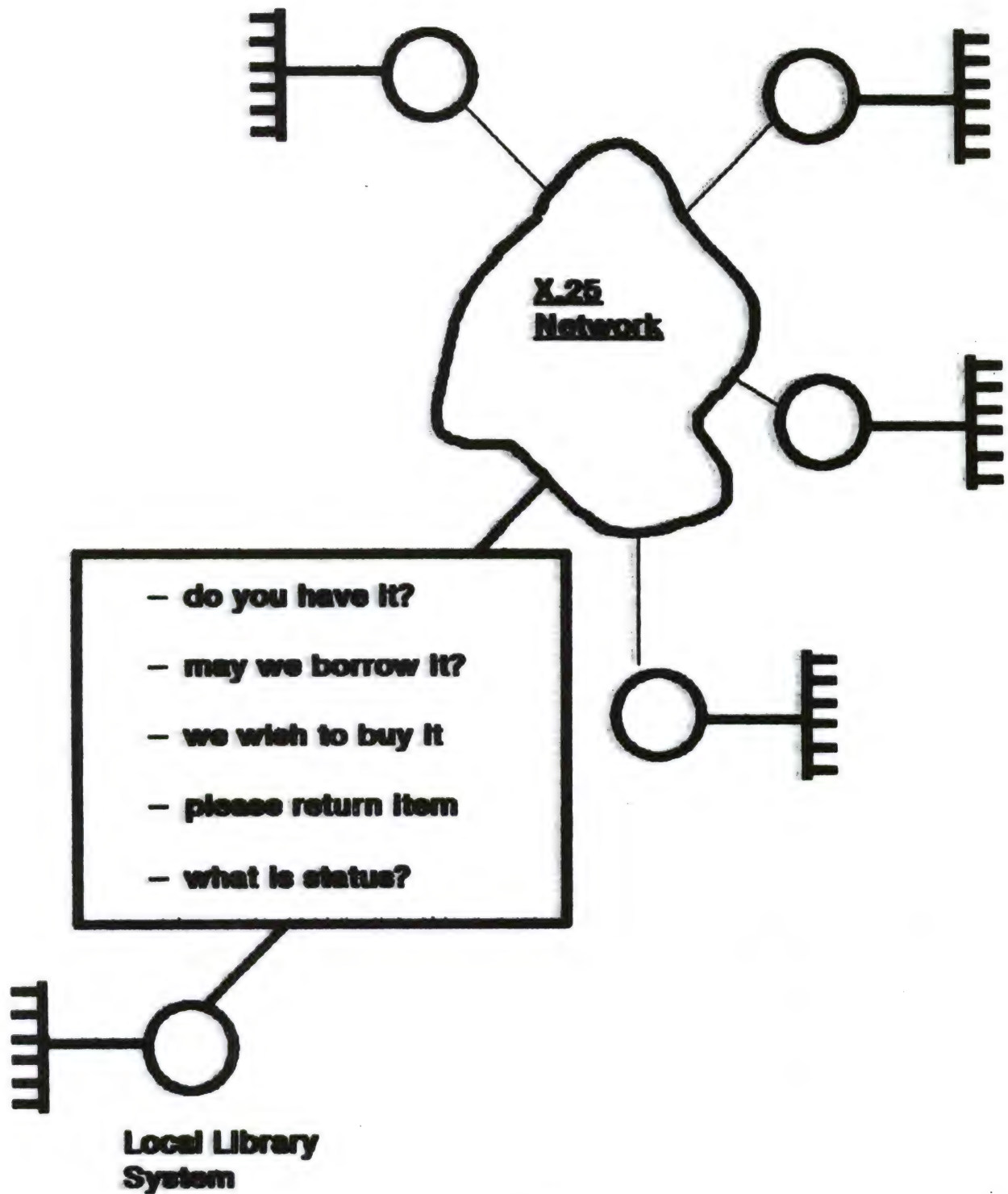
(7) Richard E. Luce, "IRVING: Interfacing Dissimilar Systems at the Local Level", Library Hi Tech, (Issue 7, 1984) pp 55 - 61

(8) Hilary Burton, et al. "Resource Sharing Through Integration of an Intelligent Gateway and Library Support Software", Special Libraries, (Winter 1986), pp 28 - 35

First LSP Production Extension

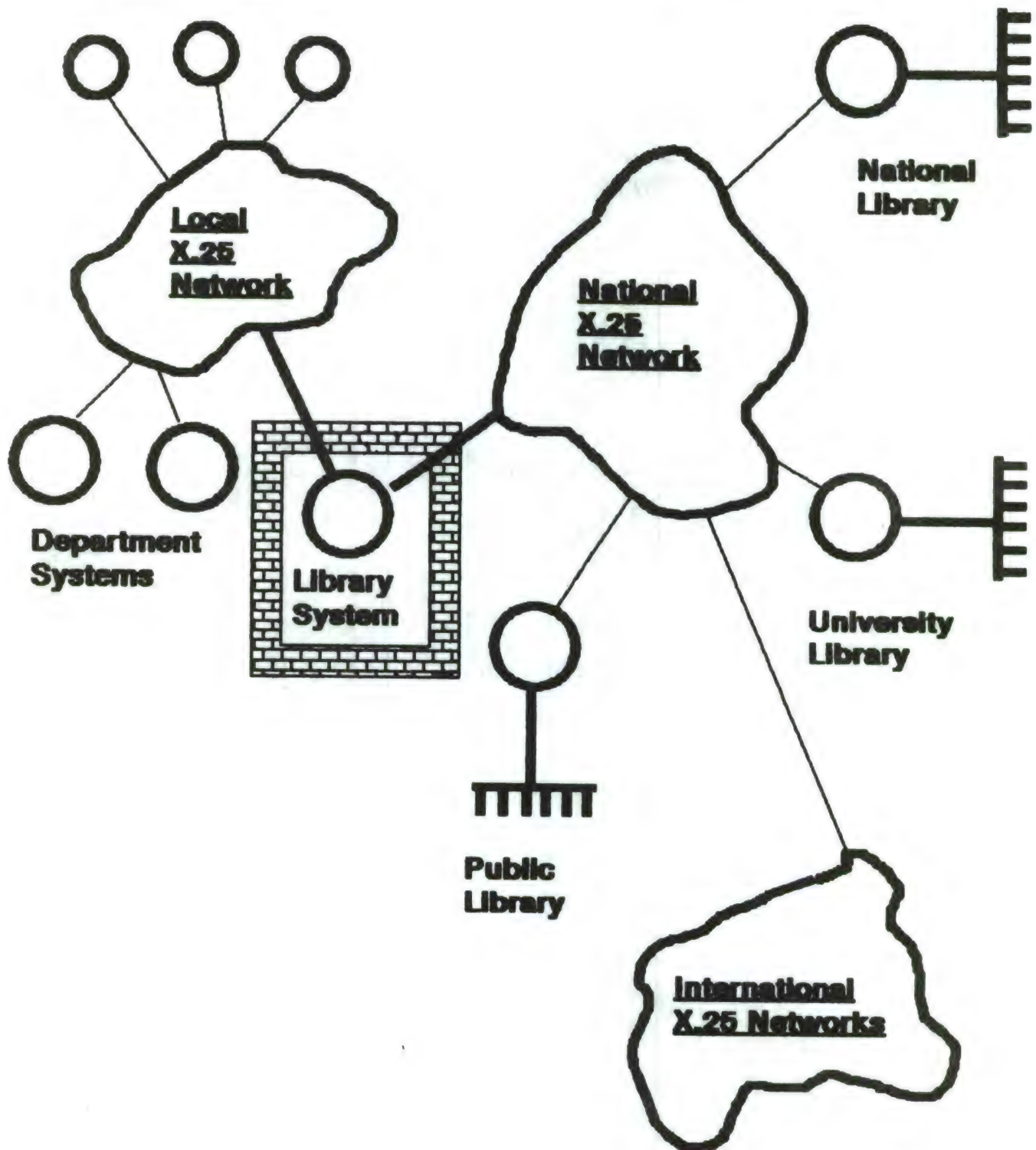


Sample Network Queries



Potential Network Model

Workstations



AUTHOR AFFILIATION: Michael Monahan is a Product Manager,
Library Systems, GEAC Computers
International, Inc.

ADDRESS: GEAC Computers International, Inc.
350 Steelcase Road West
Markham, Ontario, L3R 1B3
Canada

INTRODUCTION OF ONLINE SEARCHING TO END USERS
AT AT&T BELL LABORATORIES

Donald T. Hawkins and Louise R. Levy

We describe preliminary steps leading to the introduction of online searching to end-users at AT&T Bell Laboratories. A survey to collect user opinions was conducted; we found that the majority of potential end-user searchers have terminals, not personal computers at their disposal. We decided to offer the EasyNet system to our users, because EasyNet is a dial-up system. A trial of the system was held; users were generally favorable. This paper describes the trial and users' reactions.

Online Searching, End-Users, User Survey, EasyNet, AT&T Bell Laboratories

INTRODUCTION OF ONLINE SEARCHING TO END USERS AT AT&T BELL LABORATORIES

*Donald T. Hawkins and Louise R. Levy, AT&T Bell
Laboratories*

Keywords: Online Searching, End Users, User Survey, EasyNet,
AT&T Bell Laboratories

Abstract: We describe preliminary steps leading to the introduction of online searching to end users at AT&T Bell Laboratories. A survey to collect user opinions was conducted; we found that the majority of potential end user searchers have terminals, not personal computers at their disposal. We decided to offer the EasyNet system to our users because EasyNet is a dial-up system. A trial of the system was held; users were generally favorable. This paper describes the trial and users' reactions.

1. Introduction

Online searching services have historically been provided to the staff of AT&T Bell Laboratories through intermediaries.[1] Although many employees showed an interest in online searching, few wanted to do it themselves because it was easy for them to use the intermediary service and because the host systems were not user-friendly. With the development of front end systems, end user searching became a more definite possibility; we therefore began to investigate end user searching, with a view to introducing it to the staff of AT&T Bell Laboratories.

2. Survey

We began by conducting a brief survey of our customers to determine whether they were interested in doing their own searching and, if so, what type of equipment they had available. Over a period of about a month, the questionnaire shown in Figure 1 was distributed to Library Network customers requesting an online search. The responses to each question are shown on Figure 1. Of the approximately 125 respondents, only 15% had ever done any searching on their own, but over 80% found the idea attractive. Commonly cited reasons for wanting to do searching were speed and convenience. Among the 20% of the respondents who wished to continue using intermediaries, reasons cited were lack of time, lack of searching expertise, and the cost-effectiveness of using a professional searcher. Respondents overwhelmingly wished to use their own personal terminals instead of coming to a public-access terminal in the library. There was widespread interest in end-

Reprinted from the Proceedings of the National Online Meeting, May 1986. Copyright, Learned Information, Inc., 143 Old Marlton Pike, Medford, NJ 08055. Reprinted by permission.

user searching; over 90% of those surveyed asked to be kept informed of developments.

Since most of today's commercial online searching systems are not user-friendly and since front end software has become readily available, we sought to identify a front end suitable for the AT&T environment and conduct a trial using it. Our survey showed that many potential end users have terminals, not PC's, on their desks; we therefore eliminated PC-based front end packages from our initial considerations and focused on those that could be located on either a vendor's machine or on one of our machines. The only two meeting these criteria were IT, produced by User-Link, Ltd, and EasyNet, produced by Telebase Systems, Inc. We chose EasyNet for the first trial because it allowed us to try a front end with little initial investment. EasyNet is a dial-up front end system located on Telebase's mainframe; Telebase would therefore be responsible for software customization, updating, and much of the user support. EasyNet met other criteria we considered important for a front end. It offered access to hundreds of databases on eleven different retrieval systems; the databases were diverse enough to meet AT&T end users' needs. EasyNet is also capable of choosing the database for the user if desired; we considered that feature especially important for novice users.

3. EasyNet Trial

We purchased a commercial contract from EasyNet, which allowed us to design three customized introductory screens reflecting our sponsorship of the EasyNet service. Figure 2 shows the welcome screen. Note that a Library Network contact appears prominently. A group of limited-use passwords was also secured for distribution to the trial participants.

Approximately 25 trial participants were recruited from the survey respondents and through referrals. Users were enticed by offering them \$75 worth of searching, paid for by the Library Network. They were asked to do some searches in their fields of interest and then to participate in a focus group interview after the trial was finished. Approximately half the respondents were from development organizations; the other half represented research organizations. They were primarily seeking technical information; business information users were not well represented. All of the trial participants were experienced library users and were familiar with online searching.

Each user was given logon instructions, a password, a search log form, and a phone number to call if help was needed. No system documentation was given to the users until midway through the trial because EasyNet had none available at the outset and because EasyNet claimed the system was so easy to use that documentation was not required. Half way through the trial, EasyNet sent us a newly developed reference card; it was forwarded to the trial participants without comment from us. Participants were given approximately four weeks to use their passwords.

We had hoped to learn more about the searches done by our participants by examining their search logs. Unfortunately, we

found that the participants did not always log their searches; in fact, some participants did not log any of their searches at all. Judging from the usable search logs we received, we estimate that well over 60 searches were done by the participants.

3.1. Focus Group Interviews

At the conclusion of the trial, we conducted focus group interviews, each lasting about two hours. Open-ended questions were raised regarding the participants' experiences with EasyNet as a searching tool, improvements necessary for easier use, as well as their view of the Library Network's role in offering such a service. Figure 3 shows our interview outline.

4. Results

4.1. Need for End User Searching

The need for an end user searching capability was unanimous. It was viewed as a timesaver; participants liked the ability to retrieve information quickly and conveniently (day and night, office and home). They viewed EasyNet as a way to eliminate the trip to the library and reduce or eliminate turnaround time for their searches to be completed. Participants also felt they had more control of their searches when they did them. Most participants did not expect to be heavy users, expecting to search only once or twice a month.

4.2. Databases

EasyNet offered access to the databases needed by the participants, who were primarily interested in scientific and technical information; most believed they would need only a few databases to satisfy their queries. (Interdisciplinary and business users might feel differently.)

Users could allow EasyNet to choose the database or they could choose it themselves. Those who opted to choose the database themselves would have liked EasyNet to have an online list of available databases. (EasyNet offers a printed list of databases; because of its length, we did not reprint it for the participants.) Users wanted to get a sense of what databases were available in their subject area and their coverage. A few users, especially those who let EasyNet make the database choice, wondered what they had searched or whether their searches were thorough. Others would have liked to see at least the time span of the chosen database displayed online.

4.3. Hardware Problems

Most of the participants were using AT&T Teletype or Hewlett-Packard terminals; there were a few PC users. A major problem was that many users lacked printers; they found themselves trying to copy the retrieved citations with pen and paper! PC users did not find lack of a printer a difficulty because they downloaded their search results to disk, then displayed them later.

4.4. Overall Participant Opinion

The overall opinion of the participants was that the EasyNet system was not without its problems, but they would still use it. The general feeling was well summed up by one participant:

"This system needs a lot of work on its user interface, but I would use it anyway."

Without any documentation, users were able to navigate through the system, although all mentioned that EasyNet's reference card greatly facilitated use of the system. Users were pleased that the system was nearly always available when they wanted to use it.

4.5. Training

We discussed the issue of user training at length, particularly in view of EasyNet's claim that no training or documentation was needed to use the system, and found a clear division of opinion among our users. Those users in research areas did not want any training and were willing to take the time to experiment with the system and learn how to use it. They wanted access to the full range of databases offered by the system. About half the participants (nearly all in development areas) would like to see formal training offered. Basic searching concepts, Boolean logic, database content and scope, and system use were topics of interest.

4.6. Help Functions

Trial participants felt that help functions on EasyNet needed improvement. Online help is important for search concepts, database choice, search statement construction, and problems with search results. Some users tried EasyNet's SOS feature, an interactive human help service; most found it helpful. Most SOS queries related to database choice.

4.7. Software Problems

Nearly all participants experienced problems using the EasyNet software. Since many changes have been made since we concluded our trial, we will summarize only the major problems here.

- The most common complaint was the inability to modify or refine a search statement once it was submitted by EasyNet to the online vendor. As EasyNet was then structured, the user entered the search statement *in toto* and then allowed the process to proceed to completion without further intervention; if changes were desired, the user was required to submit another entirely new search statement.
- As soon as the first 10 results were retrieved, EasyNet disconnected from the host system; if subsequent citations or a revised search were desired, a new connection was automatically established. During the course of a search, therefore, a user might require EasyNet to redial the vendor and select the database numerous times. This was seen as time consuming, expensive, and frustrating. It also removes the interactive nature of online searching and makes browsing difficult. (We note that EasyNet now allows users to modify their searches before results are printed.)

- At the point where the logic was to be entered, the user was given the message, "Enter your search topic" A few participants did not understand what EasyNet meant by "search topic" and entered very broad general subject terms (i.e., "polymers", "numerical analysis", etc.) instead of a specific Boolean logic expression. (As a result of this observation, EasyNet changed the message.)
- Users could not abort a search if EasyNet took too long searching for an item.
- Users often found their searches retrieved too many citations and would have liked the ability to set parameters on the maximum number of records the system would retrieve.
- Users did not discover the truncation feature and therefore were unable to retrieve both singular and plural terms unless they were explicitly entered. We suspect that most users did not consider plurals in their strategies, nor were they aware of such nuances as British and American spellings.
- The list of EasyNet commands (such as "M") was buried in one of the opening help screens. If users did not read the help screens, they never found out about the commands. (We had tried to remedy this deficiency by listing the commands on our instructions when we distributed the passwords.)
- EasyNet has no SDI capability; users would like it.

4.8. Search Results

Most participants felt that the system retrieved useful information. However, search results were often too broad, and too many citations were retrieved. Users did not know how to narrow such searches. From their previous experiences with online search output, participants expected to receive abstracts, and they were not disappointed to find they still had to order the full text of the article after their search.

4.9. Pricing

EasyNet's pricing structure was the object of a great deal of criticism. (EasyNet charged \$1 for each access plus \$8 per search, where a search is defined as the retrieval of 10 citations. Subsequent batches of 10 citations are subject to another search charge. There was no charge if no citations were retrieved.) Participants felt that, too often, they paid \$8 for searches that retrieved too many items. A representative comment was, "the nerve they have charging me for the first 10 of 75,000 items". (Such large retrievals were the result of confusion surrounding the "Enter your Search Topic" message--see above.) Searches retrieving no items were often most useful. All participants agreed that the costs of services should be made known to them before using the service.

5. Conclusions

Based on the comments by our EasyNet trial participants, we have moved ahead to make the service available throughout AT&T Bell Laboratories. We see the intermediary's role growing to include consulting and training end users as well as providing information to them. One participant summed it up, "we need some kind of system, it has to go this way...but we can't get passwords, etc. on our own." Other responses by the participants indicated the need for a front end system with an interface for all levels of experience. Most users will need substantial support in choosing databases and understanding the basics of searching. Many will need only a few databases to satisfy their needs, but they need to understand the scope of the databases.

Our trial showed clearly that some AT&T Bell Laboratories employees want to do their own online searches, although most would like some help with the procedural details of obtaining passwords, etc. EasyNet allows us to implement such a service; we look forward to a new era in the history of online searching at AT&T Bell Laboratories.

6. Reference

- [1] Hawkins, Donald T., Six Years of Online Searching in an Industrial Library Network, *Science & Technology Libraries* 1(1): 59-65 (1980).

Figure 1



WOULD YOU LIKE TO DO YOUR OWN ONLINE SEARCHING?

We are studying ways to provide user-friendly access from employees' terminals to some of today's commercial online information retrieval systems (e.g. Dialog, ORBIT, Nexis, BRS, Dow-Jones News Retrieval, etc.). To help us decide what type of system would best serve the needs of AT&T Bell Laboratories employees, we need to know the interest level of our customers and what type of equipment they use. Please take a few minutes to answer the questions below and return this sheet to the person from whom you received it.

- Name _____ Department _____
- Location _____ Phone _____ UNIX address (machine/login) _____
- Have you ever done your own online searches on commercial bibliographic online information retrieval systems (Dialog, Dow-Jones, etc.)?
 Yes 13 (15%) No 74 (85%)
 If not, are you interested in doing your own online searching?
 Yes 83 (81%) No 20 (19%)
 Why or why not? _____
 - If you are interested in performing your own searches, where would you prefer to do them?
 From a terminal in my office 91 (89%)
 At a public access terminal in the library 11 (11%)
 - What type of computer equipment do you have?
 Terminal 77 (65%) What make (model)? _____
 Personal Computer 41 (35%) What make (model)? _____
 Check here if your PC is equipped with a modem. 28 (68%)
 - Would you be willing to participate in a focus group interview on doing your own searching?
 Yes 58 (62%) No 36 (38%)
 If not, are you interested in being kept abreast of AT&T-BL developments in this area?
 Yes 69 (90%) No 8 (10%)
 - Please add any additional comments you wish.

Figure 2

EasyNet Welcome Screen

WELCOME TO EASYNET

An Information Retrieval Service
brought to you by
The Library Network Operated by
AT&T Bell Laboratories

for further information or
assistance call D.T. Hawkins
(201)-582-6517
or send mail to mhuxdldthk.

Copr. 1985, Telebase Systems, Inc.

Figure 3

Focus Group Interview Outline

Mechanical improvements:

Logon-logout
Database selection
Getting output

Was the system available when you wanted to use it?
Did it find the right/useful information?
Did you look at abstracts? Were they useful?
Did you get into any full text databases?
Search cost? More than if the library did it? Too much for you?
Did you use the SOS feature?

What equipment did you use to access EasyNet?
Did you have a printer?
Did you print off anything yourself?

How did you use the output? Was it useful/helpful to you?

Should we offer EasyNet to everybody (with them paying for it)?
How much do you think you would use it?
Would you like to continue using your password (at your expense)?

Should we continue our investigations into end user searching?
What should the role of the professional searcher at AT&T be?
How should we assist users?
Would you like to know more about online databases?

AUTHOR AFFILIATION: Donald T. Hawkins is a Senior Information Technology Scientist at AT&T Bell Laboratories, Murray Hill, NJ

Louise Levy is a Market/Product Development Specialist with the Library Network operated by AT&T Bell Laboratories.

ADDRESS: Room 6A-311
AT&T Bell Laboratories
Murray Hill, NJ 07944

TERMINOLOGICAL ASSISTANCE FOR THE ONLINE SUBJECT SEARCHER

Marcia J. Bates

To the searcher, an online information system is a black box; nothing comes out of it unless the searcher specifically commands certain terms and indexes to be searched. This characteristic of online systems can be a problem for the following reasons:

1) the searcher may wish to browse and/or explore, i.e., is open to encountering information that he or she did not realize would be useful, information which, by definition, cannot be described in advance,

2) the searcher may not yet have fully articulated the query and consequently do poorly at thinking up what to tell the system to do,

3) research suggests that indexer and searcher inconsistency are so fundamental to information description and retrieval that effective search formulations must almost always contain more variety in terminology than typical end-users and even many intermediaries provide.

For all these reasons searcher queries may be inadequate to extract the needed information. Consequently, the system should help the searcher generate the necessary variety in search terminology. Various means of helping the searcher are explored, including the provision of online thesauri specially designed to help searchers generate variety. Methods of helping searchers with full text databases are also discussed.

Online Thesaurus, Online Subject Searching, Online Interfaces,
Subject Indexing, Information Search Behavior, Full Text Databases,
BRS TERM Database

1. INTRODUCTION

Most discussion about gateways at this conference has concerned programs that provide automatic logon, database selection, downloading assistance, and the like. Valuable as these capabilities are, they do not get at the heart of online searching, i.e., search formulation and modification. Research such as that of Fenichel with intermediaries [1] and Borgman [2] with student end users, as well as my own experience with bright, talented students in online searching courses convince me that effective, powerful searches are surprisingly hard to achieve.

In particular, I note that the range of talent in this area seems wider than that in other courses I teach--and does not seem to be associated with general intelligence. Some bright students do better on the first day of the course than other bright students (equally lacking in experience) do on the last day of class. In her study at Stanford, Borgman wanted her experimental subjects to reach a certain criterion level of system use capability before being included in the study proper. Stanford students are the cream of the crop of undergraduates, yet 26 percent of her subjects could not meet that minimum criterion performance level. She found that background in social sciences and humanities was more commonly associated with difficulties with online searching [2].

We in this business who write books and articles on online searching and design information retrieval systems are probably among those who have a knack for online searching. Consequently, we do not realize the full difficulty in searching experienced by some people, including very bright ones. If some intermediaries have these problems, then imagine the difficulties for end users, who know less about information retrieval generally, even if they have a knack for searching.

At the same time, while end users are performing searches that we would recognize as inadequate, they may think they are doing well, since in any good-sized database even misspelled words pull up something. The irony here may be that through the use of gateways for logon, database selection, and the like, we may make it easy for end users to get into systems which they otherwise would not tackle and which they then cannot search worth peanuts. We must aid end users with that core process of search formulation and modification--an aspect of the search process which is admittedly the most difficult in which to provide assistance.

Since the question of how to help users with these processes is difficult, I would like to draw attention to the idea that there is a wide range of possible approaches. Earlier at this conference, Linda Smith drew a distinction between systems that provide *augmentation* and those that provide *delegation*. In many cases searchers may want to delegate the entire search to the system and not have to think beyond some basic initial input, and effort is being made and will continue to be made to develop systems that allow such delegation. But there is a much wider range of possible combinations of system and searcher input in which the system augments the searcher's efforts. Computers do

some things well and people do other things well and there are a lot of interesting ways to combine these capabilities. I think we have underutilized augmentation approaches to date in this field. People do not always want to hand over complete control of the search to the system. By analogy, cameras have both automatic and manual settings for focus, exposure, etc.; automobiles have both stick shifts and automatic shifts to suit various needs. Over the long run, I expect to see information systems in both the augmentation and delegation modes coexisting and meeting different needs and personalities. The suggestions I make today will be in the augmentation mode.

One of the principal ways we can help people in the search formulation and modification stages of online searching is to make it easy to identify search terms or alter existing ones. In the remainder of this paper I will first discuss several ways in which search formulation/modification is difficult in online systems, and then suggest some ways in which searchers may be aided online in that process by the information system. The methods to be suggested all involve assistance experienced by the searcher at the interface in an augmentation mode, i.e., the system does not make internal changes out of the control of the searcher.

2. DIFFICULTIES IN SEARCHING AN ONLINE SYSTEM

Most print sources allow browsing, random scanning, serendipitous discoveries. Current online systems permit browsing only in a very limited way. The "Expand" or "Root" capabilities require the input of some initial word, and even then the searcher does not get to see much--at least not in comparison to the amount of information a person can take in at a glance when looking at a page of type, or when flipping through pages. To the searcher, an online system is a black box. It will not do anything until you tell it what to do. There is no way to "see inside" the black box to see what is there, without telling it what you want in advance.

Both anecdotal and research evidence have long shown the importance of unexpected information discoveries for researchers. We may say that people need two kinds of information, the kind they know they need, and the kind they do not know they need, but recognize when they see it. Both types are of great value, and it is essentially only the first type that is available in most current online systems. Human curiosity, expressed in exploratory behavior (i.e., browsing in information systems), is not wasteful behavior. It is biologically adaptive [cf. 3], because such random behavior exposes the individual to new, unanticipated information, some of which will promote the survival of the individual. The same can be said for unanticipated information related to one's research work. Thus, in the long run, we must make true browsing possible in online systems, or they will remain inferior to manual sources in at least this respect.

Another reason why we should overcome this black box feature of online systems is that people often have difficulty getting clear in their own minds what information they want, or articulating the need well even when they do know what they want. Belkin has argued this point very persuasively [4]. At UCLA I have had the

opportunity to examine some information request sheets filled out by end users before they have an interview with the online search intermediary. I have been impressed by the number of educated end users who confuse the search process in one imaginative or bizarre way or another. Expressing an information need is not a simple, automatic, or easy process. People may need more help in query formulation than we have recognized.

A third reason why we should overcome this black box quality has to do with some fundamental features of human cognition and of the search process. In a paper of mine appearing in the *Journal of the American Society for Information Science* [5] I argue from a review of empirical research that both selecting index terms by indexers and selecting search terms by searchers is a partially indeterminate and probabilistic process. For example, one study was done in office automation where the researchers wanted to find the most common terms used for editing functions so that they could give easy-to-remember names to the editing capabilities on a word processing system. They gave experimental subjects a corrected manuscript and told them to say what they would tell someone else to do to correct the manuscript if the other person could not see it. One would expect these common editing functions to have just one or two names each. Instead, the researchers found that "the average likelihood of any two people using the same main content word in their descriptions of the same object ranged from about .07 to .18" [6].

There is thus enormous variety in terms used by people for the same content. We do not know how to predict which term someone will use or why. This same pattern of great variety is found in studies of indexer and searcher consistency, as well as in psychological association research. The implication here is that where there is so much variety in document description, there must also be great variety in use of terms in search formulations or else relevant materials will be missed by searchers. If a certain topic or area is described under a dozen terms, then a searcher must use the dozen to find all relevant documents. But recall that we are dealing with a black box. The searcher must produce all the necessary terms; assistance by most online systems is limited.

3. TERMINOLOGICAL ASSISTANCE TO THE SEARCHER

Operationally, a very important part of assistance with the above-stated problems consists of help with the selection of terms. Showing searchers a network of terms and term relationships will suggest novel ideas for browsing, help people decide just what they do want in their search, and suggest the many additional terms they may need to capture the full variety of terminology describing the desired information.

A fundamental change that is required is that the system, in response to minimal initial input, "spontaneously" produce information that will help the searcher proceed with the search, rather than wait for the searcher to tell the system exactly what is wanted. To make that possible, the system must, in turn, have on hand information about term and document relationships so that

related terms, classification hierarchies, example documents, or other terminological information can be given the searcher to help articulate or expand a search.

The best option, I believe, is to develop a pattern where information is flashed on the screen and the searcher may select desired terms or citations by some simple mechanism like keying in term number or clicking a mouse cursor on the term. Any information of no interest to the searcher may simply be *ignored* and will disappear as the search moves on. Thus the system is constantly providing additional information, while at the same time not requiring the searcher to do anything about that information other than ignore it.

Such an arrangement allows searchers to take advantage of two human capabilities which are not well used in current online systems: 1) recognition as opposed to recall ability, and 2) ability to scan or take in at a glance. With respect to the first capability, it has long been recognized in psychological research that human beings can recognize far more terms than they can recall at any given moment. A system which provides related terms or example document references can help the searcher greatly enrich a search formulation beyond what spontaneous recall would permit. With respect to the second point, people can often tell at a glance whether there is any information of interest on a page or screen containing a great deal of information that would take a long time to read straight through. Using this ability to scan or glance, a human being can review a lot of information of marginal utility very quickly and pick out that serendipitous fact or reference which may prove of great value, and which is essentially unpredictable in advance. Thus, a generous online system will provide a lot of information to searchers, much of which will be ignored, but some considerable portion of which the searcher will want and would not have known to ask for.

4. ONLINE THESAURI

A general approach for assisting the searcher was suggested in the previous section. In this section we will consider some specific changes that might be made in online thesauri. A variety of such changes are suggested; it is not clear yet just what design will be most helpful--perhaps all of these approaches can be used in one way or another. The possibilities are discussed below under four headings, 1) searching on descriptors: end user vs. indexer thesauri, 2) special characteristics of online thesauri, 3) special characteristics of full text thesauri, 4) "front-end system mind" for online information systems.

4.1 SEARCHING ON DESCRIPTORS: END USER VS. INDEXER THESAURI

Existing thesauri are designed primarily for indexers and only secondarily or at all for intermediaries and end users. I call these "indexer thesauri." Such thesauri are intended primarily to show indexers the legitimate, or allowable terms with which to index documents. The searcher must in turn use the same descriptors to find documents (if searching on controlled

vocabulary). Indexer thesauri often list terms not actually indexing documents in a given system or institution at a given time, or fail to list terms or combinations of terms actually indexing documents. For example, all proper noun terms in use, such as names of individuals or places, are seldom included in thesauri. Trained indexers understand these quirks; end users seldom do. Indexer thesauri may use codes such as "BT" which are mysterious to the end user, or make distinctions between legitimate and illegitimate (e.g., "see from" terms) which are not clear to the end user. Finally, and most importantly, most indexer thesauri have only a limited number of entry terms, that is, terms which are not legitimate indexing terms, but which are listed so the searcher may be directed to the terms that are legitimate for indexing (i.e., descriptors).

Indexer thesauri may be contrasted with what I am calling end user thesauri, that is thesauri designed primarily for the searcher, especially the end user. The latter do not exist for the most part currently, though some features that are more searcher-oriented are beginning to appear in existing thesauri [7]. An end user thesaurus, among other features, would list all the terms currently indexing documents in the system and use self-explanatory codes and cross-references. The variety in initial search terms (i.e., entry terms) likely to be used by searchers is enormous--we may safely guess that it is vastly greater than the variety to be found among legitimate terms--so a good end user thesaurus should also contain a vast number of entry terms, which then lead, through pointers, to legitimate terms. Thus the searcher using an online version of such a thesaurus should be able to enter any term, legitimate or not, and be shown the conceptually closest legitimate term as well as related descriptors to use in searching.

4.2 SPECIAL CHARACTERISTICS OF ONLINE THESAURI

Online thesauri to date have tended to be simply manual thesauri mounted in online form. Such an approach fails to take advantage of both the special capabilities of online systems as well as the special needs associated with online searching. It is time we considered the possibility that effective online thesauri may be very different creatures from their manual counterparts. In this section we will take a different angle on the end user thesaurus idea.

The BRS TERM database, an online thesaurus, represents an important departure from traditional thesaurus design, being both geared more to the searcher's rather than the indexer's needs, and containing features more suitable specifically for online searching than most traditional thesauri. It has at least three features which distinguish it from traditional approaches. I recommend that we use these features more widely in online thesauri, and that terms from such a thesaurus be shown automatically to the searcher in the early part of a search. Those features are the following:

1. The thesaurus contains several merged vocabularies. The purpose of a traditional manual thesaurus is to show which terms

are legitimate in a given catalog or manual database for indexing or consequent searching. But in an online system the searcher is no longer restricted to legitimate index terms; free text terms are excellent search terms too. Thus, as online searchers have been discovering, one may use any thesaurus as a source for free text search terms, as long as it suggests terms in any way useful for a given query. By merging several related thesauri, the TERM database nicely supports that broader need for terminology in online searching.

2. The TERM database also suggests free text terms, i.e., terms not listed as legitimate in any of the merged vocabularies but which the database creators believe may be useful for free text searching. These features of TERM are making clear that with online searching one of the traditional purposes of thesauri--to show which terms are legitimate--is fading in importance in comparison to another purpose: to show the full variety or range of terms that may be used to search a topic.

3. Clusters of terms to be ANDed together are displayed. In point 2 above, the wide range of free text and index terms represent potential Ored sets for searching in a system using Boolean logic. The TERM database also shows clusters of terms that may be ANDed together to form a Boolean search statement for topics that are composed of more than one distinct concept. Such is an entirely reasonable and needed feature of any thesaurus designed to assist in a system permitting Boolean searching.

4.3 SPECIAL CHARACTERISTICS OF FULL TEXT THESAURI

We are just beginning to explore the characteristics of online searching of full text databases. It is time also to begin thinking about how online thesauri for full text searching should be designed. My suggestions here are first thoughts only on this interesting question.

Features discussed in the two previous points are needed even more so for full text searching. Index terms and title terms--the traditional terms most commonly matched on in bibliographic databases--are more highly structured and limited in variety than the full text of the article or book itself. With full text searching we drop down into the full variety of natural language itself, with an attendant need on the part of the searcher to generate great variety in order to find relevant passages.

The thesaurus needed now begins to approach the meaning of "thesaurus" in the original sense as used by Roget. Such a thesaurus shows clusters of closely related natural language terms. Now the question of legitimate vs. illegitimate terms disappears altogether (except where the full text is also given descriptors). The searcher would use such a thesaurus somewhat differently than the user of a Roget's thesaurus would, however. In the former case one wants to examine and select many terms for searching; in the latter case one is seeking to find the one best term to express one's meaning.

There are a couple of other differences as well between a full text online thesaurus and a traditional Roget's thesaurus. For online searching, one needs terms at different levels of

specificity. For example, the searcher interested in "newspapers" as a topic should also search on narrower terms such as "Los Angeles Times" and broader terms such as "media." Thus, a full text thesaurus should have clusters of terms that explicitly include terms at other specificity levels or which direct the user to clusters at those other levels. Secondly, online full text searching will almost certainly require a great deal more technical vocabulary than appears in traditional Roget-type thesauri.

4.4 "FRONT-END SYSTEM MIND" FOR ONLINE INFORMATION SYSTEMS

The points made so far have dealt with variations needed in thesauri in order to convert them to effective online aids to online searching. We may take this approach a step further and suggest a still more comprehensive aid to the online searcher in the system interface. Elsewhere, in discussing online catalogs [5], I have described a "front-end system mind" (FSM for short), which is a network of terms and term relationships. There is an almost unlimited number of different kinds of relationships which may be represented in such a network--related terms, broader and narrower terms, terms related in classified categories, terms related through co-occurrence in documents, and so on.

This FSM is made into an effective interface and search aid by adding capabilities which enable the searcher to press keys for "Action codes". For example, the searcher may click a term, then press an action code number which stands for "Show me terms related to this one", or "Show me example documents in which this term appears", or "Show me the definition of this term", or any of a half dozen other such commands. I see no reason why such an FSM could not be designed for online bibliographic or full text databases as well.

5. SUMMARY

A variety of difficulties for searchers in formulating and modifying search queries have been discussed, and a variety of means for helping searchers overcome those difficulties have been suggested. Attention has been focussed on means of improving online thesauri, both for conventional bibliographic databases as well as full text databases.

6. REFERENCES

1. Fenichel, Carol Hansen. "Online Searching: Measures that Discriminate among Users with Different Types of Experiences," Journal of the American Society for Information Science. Vol. 32, no. 1, January 1981, 23-32.
2. Borgman, Christine L. "The User's Mental Model of an Information Retrieval System: An Experiment on a Prototype Online Catalog," International Journal of Man-Machine Studies. Vol. 24, no. 1, 1986, 47-64.

3. Bates, Marcia J. "An Exploratory Paradigm for Online Information Retrieval," Proceedings of the 6th International Research Forum on Information Science, Frascati, Italy, Sept. 1985. Amsterdam: Elsevier, in press.
4. Belkin, N.J. et al. "ASK for Information Retrieval: Part I. Background and Theory," Journal of Documentation. Vol. 38, no. 2, June 1982, 61-71.
5. Bates, Marcia J. "Subject Access in Online Catalogs: A Design Model," Journal of the American Society for Information Science. Vol. 37, no. 5, in press.
6. Furnas, George W. et al. "Statistical Semantics: How Can A Computer Use What People Name Things to Guess What Things People Mean When They Name Things?", In Association for Computing Machinery, Proceedings of the Human Factors in Computer Systems Conference, March 15-17, 1982, Gaithersburg, MD. New York: Association for Computing Machinery, 1982: 251-253.
7. Piternick, Anne B. "Searching Vocabularies: A Developing Category of Online Search Tools," Online Review. Vol. 8, no. 5, 1984, 441-449.

AUTHOR AFFILIATION: Marcia Bates is an Associate Professor in the Graduate School of Library and Information Science at the University of California at Los Angeles.

ADDRESS: Graduate School of Library and Information Science
120 Powell
University of California at Los Angeles
Los Angeles, CA 90024

Developing System Interfaces for an
Intelligent Gateway to a Heterogeneous
Resource Environment

Hilary D. Burton

This paper discusses the multiple system interfaces to the various resources available on the Technology Information System (TIS) Gateway developed at the Lawrence Livermore National Laboratory. These include available scripts and menus which are customized to the subset of resources for specific user groups. More generalized menus are also available for use by heterogeneous audiences.

Gateways, Intelligent Gateways, User Interfaces, Technology
Information System

There are multiple system interfaces to the various resources available on the Technology Information System (TIS) Gateway developed at the Lawrence Livermore National Laboratory. They include tutorial scripts and menus which are customized to the subsets of resources selected for specific user groups. They also include more generalized menus for those resource sets accessible by more heterogeneous audiences. However, the primary characteristic of the Gateway is its function as a virtual information resource. As such, it has many target audiences and an almost unlimited set of external resources which it might access. The breadth of the potential user community and the comprehensiveness of the resources we wish to make available make it difficult to accept the idea of a single user interface. This is especially true since one of the strengths of the TIS Gateway, which utilizes the RTI Ingres Relational Database Management System, is the ability to install "views" of resources tailored to each user community.

However, for the generic view of the Gateway - that version which is used by the most widely varying groups of users - there are very real needs for several levels of interfaces. The current work involves developing these interfaces for very specific applications - such as command translators between identified on-line systems - and then adding modules to make these interfaces more generally applicable. There are several reasons for this approach: 1) There still is no universally accepted standard or ideal in spite of all the work which has been done in Europe and the United States. 2) Different approaches seem to be successful with different user groups and different levels of expertise. 3) The rapidly changing spectrum of on-line systems makes it questionable to assume that what works for today's systems will work for tomorrow's.

In 1980, the following statement was made:

"The importance of specific requirements varies according to the sort and purpose of the systems (...), and to the type and knowledge level of the users (.....). It is much easier to design a system that will be operated by a homogeneous group of people having equal knowledge, training, and practical experience, than a system for a heterogeneous population of users."¹

1. Stibic, V., "A Few Practical Remarks on the User-friendliness of On-line Systems. J. Information-Science, Vol. 2, 1980, (p.277-283)

Not only is the statement equally true today, but today we are dealing with the need for interactive interfaces to multiple types of systems - bibliographic, full-text, numeric, diagnostic, predictive, etc.

We have adopted an approach which we hope will provide us maximum flexibility - both to accommodate changing user needs, but also to allow for the development of unforeseen types of information systems.

The general structural approach of the TIS Gateway is based on a categorization of resources and presentation of these categories via menu to the user. By resource, I mean a system such as SDC or BRS; or a software tool, such as SIRE or a word processing package. Resources may either be locally available on the Gateway host computer or may be located remotely. A growing number of resources that we access are located outside the United States.

At the most general menu level, the menu categories are based on user audience; for example, the FAA, DOD, DOE Cost Estimators, etc. Below this, the target resources are grouped into seven general classes. One example which demonstrates the breadth of the resources accessed is the category, Integrated Computer Resources. It is further divided into directory information, bibliographic information, numeric information, general information (such as CompuServe or the Source), computer facilities, news services, and communication services (such as Tymnet or Telenet).

In the near future, we plan to implement a combined AI/directory function which will provide the user with more useful information to make his way through the system. Instead of simply viewing a menu which includes the various classes of resources, he can begin by specifying a general or specific area of interest and be assisted in his navigation to the appropriate resource.

Currently, within a group or category of like resources, the treatment is broken into three sections. The first section is a description of the resource - what it offers, how it operates in general terms, some idea of what the cost to use it is, where it is located, and a contact point including address and telephone number, when these are available. This section is designed for the user who may never have heard of the resource or who may have heard of it, but has no knowledge of what it actually does.

The second section provides the detailed instructions on how to actually use the system. In many cases, this section consists of the service's own on-line tutorial or training information downloaded to our disk so the user incurs no connect time charges while reviewing the information. Where we feel the service's tutorial is either too lengthy or too difficult for the user to deal with, we develop our own script but include instructions or guidance for obtaining help or training in the host system. For the many specialist-oriented systems, such as a chemical data system or mathematical modeling system, we indicate to the novice user that he might wish to contact a designated office or individual for preliminary assistance.

Connection to the resource is accomplished by the third section. The user can either issue a "Connect" command with the appropriate host name or he can select the connect option from the menu and simply enter the menu number. The Gateway then chooses the optimum path to access the resources, choosing among Tymnet, Telenet, direct dial, etc., and defaulting to lower speed paths when the high speed routes are not available. Log-in is accomplished and the user is deposited in the system, ready to start searching in the native language. Passwords and accounting information are specific to the user's account. Users can log into systems with group passwords or use their own special accounts.

At the point of connection to the target system, the user may issue a single command which will open a local file and save a complete transcript of the subsequent session including both the user's as well as the system's dialog. For many of the resources currently accessed, we have developed a series of routines for further analysis and processing of any information acquired in the session. These are the post processing routines mentioned by several earlier speakers.

In most cases, to use the target resource, once a connection has been established, the user must know the system's language. However, an alternative is available. The Easynet service is available as a selection from the menu which accesses general computer-based information services such as CompuServe or the Source. In fact, since CompuServe now offers access to Easynet via their Iquest option, there are actually several ways to access this resource. In one sense, this alternative, Easynet, should serve to alleviate the pressure to develop an interface for the truly novice user (and generally one who has no desire to become anything else). However, it does not satisfy our need to develop an interface across systems used by information specialists who may wish to use

the native language of two or three services, but who must deal with five or more systems on a recurring basis.

In a project which has been anticipated for at least two years, but which is only now making any real progress, we are developing; first, a translator across six on-line retrieval systems, and second, a common command language which will provide another option to the user. Command translation will be available across DOD/DROLS, NASA/RECON, Lockheed DIALOG, SDC's ORBIT, BRS, and DOD's MATRIS system, an on-line retrieval system which features technical reports on DOD-relevant manpower training. Because MATRIS operates using the BASIS database management system, the new command translator will be capable of handling the DOE/RECON system after its transition in 1987 to a split between commercial systems like Lockheed DIALOG and the reduced in-house operation which will be managed using BASIS. The current translated Gateway common format to which all downloaded data is converted had been a DOE/RECON format. This cumbersome format will be dropped and a more clearly formatted and bibliographically standardized format substituted.

In the first phase of the translator project, we have selected the DOD/DROLS and DIALOG languages for translation. Obviously, these two represent extreme variations in implementation. Although developed at about the same time in the 60's, DROLS is noticeable terser than DIALOG, and DIALOG has undergone steady modification and enhancement over the 20+ years of its existence. Furthermore, DIALOG represents a fully developed language which supports far more than the minimal functionality identified in most command language studies. Conversely, the DROLS language was originally designed for the Defense database and has not been generalized in its scope. However, completion of the translation function between these two should provide an excellent foundation for developing many additional translators. Eventually, a user will be able to use the command from one of the six target systems to any of the other five.

Beyond the translation function, however, arises the question of placement of the language management function. For the six systems we have identified, it is a straightforward job to locate the translation capability for the user. But when dealing with a generalized command translator, placement of this function becomes more complex.

The TIS Gateway is a UNIX-based system which is table driven at even its most basic level. This gives it tremendous flexibility and capability. However, many users do not care to exercise such power, and as with UNIX, would prefer some type of screen to mask them from the specifics of the technology. The capability to allow users to shift modes between system and user control which has been referred to as a "mixed initiative" and discussed by Marcus is not easily implemented. "Exactly how such an integration and mode control can be achieved appears to represent the frontier in this area of research".²

Presenting the user with a direct choice and assuming he will be consistent (or suffer the consequences) in whichever language he selects, the translation can be handled rather straightforwardly. However, it becomes a much more complex problem if we 1) overlay a translation capability to take correct or incorrect user's commands; 2) phrase the command appropriately for the host system; 3) rephrase it for a second system; 4) allow the user to modify it in whatever his own language is and 5) then send it - keeping it separate from commands to the Gateway itself. Throughout this activity, it is desirable to allow the user to easily (unobtrusively) revert to the host native language.

For example, ProSearch and Scimate, two of the most popular front-ends to multiple host systems, access limited numbers of systems (DIALOG, BRS, SDC, NLM, and Questel). And, in the case of Scimate, the front end can search only the bibliographic files of the hosts. Furthermore, to take full advantage of the host services, it is recommended in their documentation that the user revert to native language rather than use the overlaying command language which serves as a filter and therefore cuts out certain layers of responsiveness and control between system and user. Certainly, the fact that Easynet, which started out completely menu-controlled and removed direction of the retrieval process totally beyond the user, has developed a user-driven database selection, native mode search language option says something about the shortcomings of even our most popular command command language implementations.

2. Marcus, R.S. "An Experimental Comparison of the Effectiveness of Computers and Humans as Search Intermediaries. J. Amer-Soc. Info Science 34(6): 381-404. November, 1983

For these reasons, at Livermore we will be paying considerable attention to the mixed initiative mode, for the placement and movement of the language management function, and we will have as a major objective, the development of modular tools upon which we can build as new types of retrieval systems become available, as new query languages gain acceptance, and as our user community expresses new needs.

The technology certainly exists to solve our problems - but as is true in so many cases, our understanding of the interaction of the parameters and, in this case particularly, the extension of heterogeneity in several dimensions: user population, host resources, and Gateway functionality - have slowed our progress. But, having defined an approach which should minimize any wasted effort, and provide a basis upon which we can continue to accommodate new variables, we expect to proceed.

AUTHOR AFFILIATION: Hilary Burton is a project manager for the
Defense Gateway Information System at the
Lawrence Livermore National Laboratory

ADDRESS: University of California
Lawrence Livermore National Laboratory
P.O. Box 808, L-1 512
Livermore, CA 94550

ISSUES RAISED BY COMMON COMMAND LANGUAGE STANDARDS

Dineh M. Davis

Current advances in computer technology and applications have created the right environment for exploring new issues related to the use of the Common Command Language (CCL). Although the classic objections to imposing standards on the computer industry still remain, as the use of information retrieval systems expands beyond the high-support environments of libraries and information centers it behooves us to look at alternative or additional applications for Common Command Language standards. Two levels of use may be suggested: (1) identifying the intent of the information seeker and allowing for the option of conducting a process-oriented search, and (2) using CCL primarily at the design level, making it transparent or optionally transparent to the user at the interface level.

Standards, Common Command Language

ISSUES RAISED BY COMMON COMMAND LANGUAGE STANDARDS

To put my ideas about computerized information retrieval systems in perspective, I have chosen to view the information seeker as a traveler and the path to the desired information as the journey. Just as some of us travel with the goal of getting somewhere while others travel for the sake of the process, seekers of information can also be viewed as goal or process--oriented. My contention is that the majority of interfaces designed to date are relying on the highly goal-oriented user who will in turn be most satisfied with a standard Common Command Language. However, this interface can inherently lead the information seeker toward self-fulfilling prophecies of what must be gained out of every given journey. On the other hand, the process-oriented traveler or the one whose goal is not crystal clear at the beginning of a journey is left out in the cold, with little to guide him or her through uncharted territories.

As with any other temptation to categorize people, it is imperative to keep in mind that most people are hybrids and that each of us has qualities or needs that overlap the boundaries of the divisions provided in this model.

The Traveler - For the sake of simplifying this model, I am merely looking at four general types of information seekers or travelers: Seasoned, Comfortable, Uncomfortable, and Armchair. I will not concern myself with those who not only lack the desire to travel, but don't even care to learn about the territory.

The Territory - Knowledge of information retrieval systems in general can affect the manner in which individuals will direct their activity. One can be New to the territory, Familiar with the terrain, or operating out of Home Turf.

The Purpose - The information seeker may be traveling for Business, for Pleasure, or for Business and Pleasure combined. We must be conscious of the fact that many individuals approach their Pleasure in strictly business terms, while others reap much pleasure from their Business. Yet, today's information retrieval systems are built uniquely for the business-minded. Our only means of catering to the needs of others is through the judicious selection of an appropriate database, but the means of searching this database is no different from searching any other database (assuming that both are text- or bibliographic-based).

The Know-How - Subject expertise and/or database expertise add another level of complexity to the many variables to be considered in this journey. There are those who are Just Learning the Ropes and there are the Old-Timers; with many gradations in between.

The Need: As already mentioned above, our traveler can be Focused, with an articulated goal or s/he can be Cruising or process-oriented.

The Help - Many travelers/information seekers are do-it-yourselfers. They may take longer to reach their goals, but it may be that the process of reaching that goal is just as important to them. They seek active participation while many others prefer guided tours. Of course, the Armchair travelers will rely most heavily on a human intermediary and are quite happy just to get the results in a suitable form.

Methods & Tools for Interfacing. Some prefer animate interfaces (the travel agent and tour guide; the librarian or intermediary), while others will opt for inanimate aids (the tour book and map; the online help, user manuals and reference books). Some wish to be directly involved in the search process, albeit in the presence of the intermediary, while others approach the task indirectly and can be satisfied by leaving the physical search process to another.

Their methods for seeking the appropriate path can fall into three categories or a combination of the following:

Conventional - such as using the tried and true systems which rely on command languages, menus, and modified natural language interfaces to access text or numeric databases.

Non-conventional - such as using integrated image/text/numeric databases with natural language interfaces or other new aids which are based on substantial bodies of existing research. (See, for example, Bergman and Smith "Dealing with Heterogeneity: An Architecture for Uniform Access/Integrated Presentation," this volume.)

Unconventional - such as using unusual interfaces based on new research whose results cannot necessarily be substantiated through past usage. (See, for example, Pincus and Pincus "Functionally Intelligent Interface," this volume.)

Two underlying considerations in all of the above

categories are time and money constraints. Both the world traveler and the humble information seeker have been known to modify their quest based on the ready availability of these resources. Just the same, looking within a universe that presumes the availability of time and money, permutations of the paths considered above lead to over 2300 different ways that individuals can seek - but not necessarily find - information through a computer-based information retrieval system (you can follow one such path through the travel analogy on the next page)

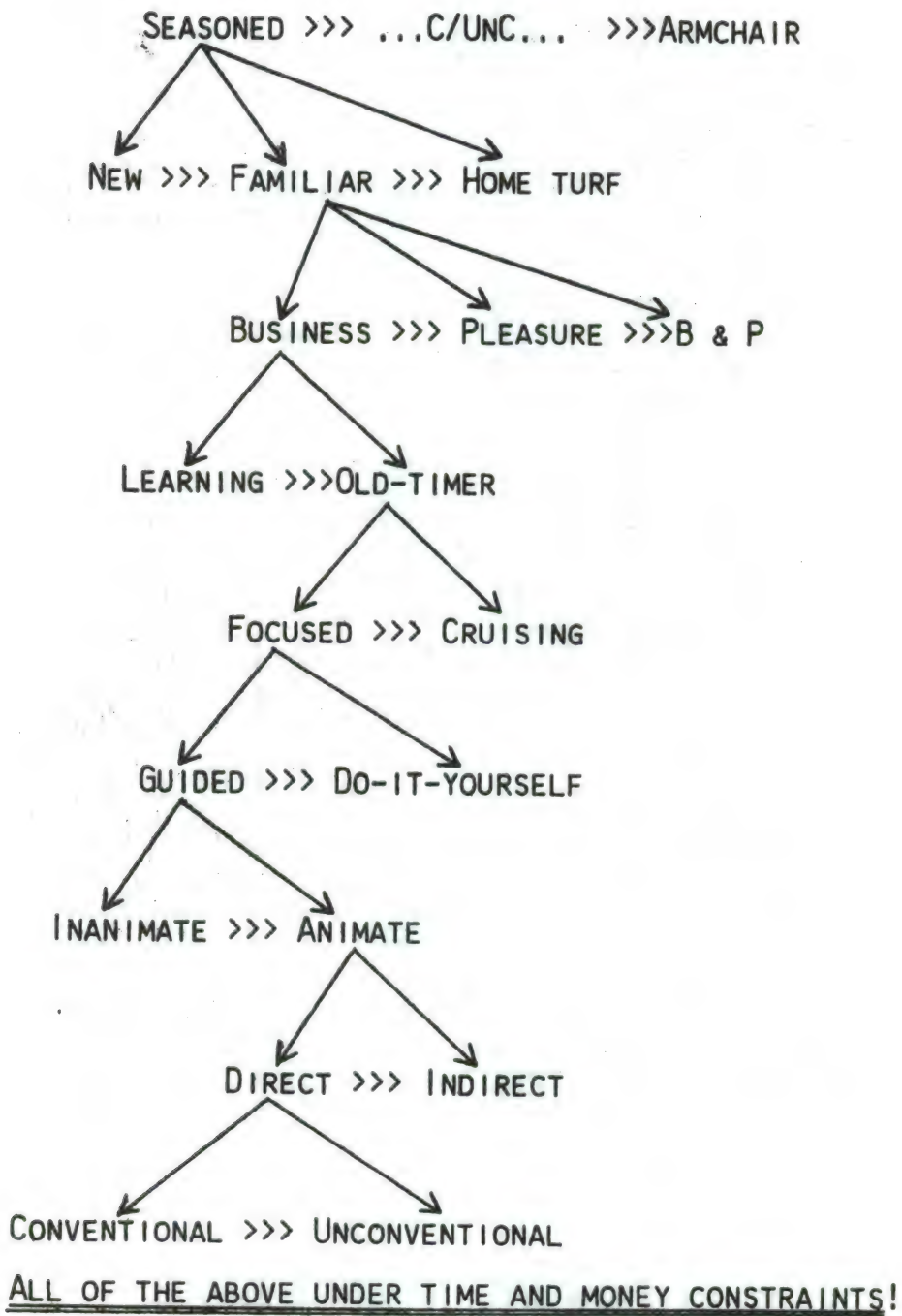
Standards

Where have we tried to apply standards to information retrieval systems? Wherever we have had the opportunity to impose controls. Where we were unable to standardize our user population, we substituted a trained intermediary who became a standardizable entity. Where we were unable to deal with process-oriented queries, we standardized the questions on our interview sheets: BE SPECIFIC; USE KEY WORDS. Where we could not standardize the user's query process, we offered only menus. And where humans are objecting to the confusion of too many commands for the same process, we can offer a Common Command Language. Yet, at this same juncture, we could also offer gateways and natural language interfaces - and it is being done.

Two prerequisites to understanding standardization in the field are the acknowledgement that we can only hope to standardize what is under our control and that standards are user-driven but users are not standard-driven.

By the first statement I mean simply that unless we have direct control over a process or an industry, we cannot impose our will to standardize that process or that industry. My second point is that in our American culture there is such a high premium placed on individuality and individual rights, and on our ability to be creative and unique, that it is often self-defeating to try to impose standards. Yet, it is these same unique and creative individuals who must band together to form committees and organizations to try to impose standards on an industry that is either incapable of reaching a standardization consensus or does not believe that one is necessary. By the time users become heavily involved in the products of the computer industry, this dilemma generally leads to standards proposals that are reduced to being "too little, too late" because of the fast pace of technological developments that render them obsolete.

VARIABLES FOR TRAVELING TO A BOSTON CONFERENCE



Such may be the case for the Common Command Language in the face of natural language interfaces and gateways of the mid to late 1980s. The information retrieval technology of the late sixties and early seventies gave rise to most of the major services currently available on the market. The real constraints of hardware and software at that time imposed certain restrictions on interface design. We are no longer plagued by those same constraints, yet the interfaces remain as the familiar old shoe that still fits.

Information retrieval from computer-based systems has reached a stage reminiscent of the end of the closed-stack era in libraries. For hundreds of generations the card catalog - and before that the accession list - served as the librarian's aid in finding material in the closed stacks. Eventually the librarian had to share this access method with the readers. Still, the tool was not intended to be entirely self-explanatory and even today we find that many library users must check in with the reference librarian once their question takes them beyond author, title, and a straightforward subject. Unless, of course, they can go directly to the stacks to browse through the collection for themselves. The open stacks have relieved much of the pressure from the librarian and the card catalog.

We must prepare ourselves to learn a lesson from the well-established profession of librarianship with its years of experience in information retrieval. We must allow the users of computer-based information retrieval systems the option to use alternatives to the "card catalog" of online databases: the Common Command Language and the thesaurus. As much as card catalogs are always going to be a necessary and integral part of any large library, we are also likely to need a standard access method to online databases. Yet, there are many individuals who also seek knowledge by walking into the open stacks of a library to browse at will. For them, let us open our computerized files and let the travelers seek their own unique ways of navigating the territory.

AUTHOR AFFILIATION: Dineh M. Davis is an Assistant Professor,
Computer Information Systems Department,
Bentley College, Waltham, MA.

ADDRESS: Computer Information Systems Dept.
Bentley College
Waltham, MA 02254

THE COMMON COMMAND LANGUAGE

Emily Gallup Fayen

A Common Command Language has been proposed for use by systems designers and those developing new user interfaces. While controversy has raged over the appropriateness of a standard command language and those who are working in artificial intelligence believe that natural language interfaces will make it unnecessary for users to learn such a language, systems developers and those working on gateway systems welcome a standard command language, because it will make system linkages easier both for systems developers and for users. The draft Command Language Standard proposed by the NISO Z39G Committee is presented with comments on the approach and the resulting language.

Common Command Language, Standards, NISO Z39G Standard Committee

The need for a common command language first began to be discussed in the late 1970's and early 1980's when the number of online catalogs and remote database services had increased to the point where it was becoming difficult for even a well-trained online searcher to keep all the commands and syntax straight. Systems developers were less enthusiastic. They urged that a move to standardize too soon would hamper development and that no one would follow a standard anyway. However, as new systems have continued to appear and older ones have evolved and changed to meet public demands, some consensus is emerging with respect to the features these systems must support and a search dialogue that works well for users. In addition, there is great interest in linking systems. The Linked Systems Project, or LSP as it is called, is a program that will support transfer of bibliographic and authority records among the Library of Congress (LC), the Research Libraries Group (RLG), the Online Computer Library Center (OCLC) and the Western Library Network (WLN). There is also great interest in providing gateways that will allow users to connect to multiple remote systems running in different environments using a common command language.

The NISO Subcommittee G was established to specify a common command language for use with databases. The committee was chaired by Charles Hildreth (READ, Ltd.). Members of the committee are Wilhelm Bartenbach (H.W. Wilson Company), Harry F. Boyle (Chemical Abstracts), Emily Fayen (University of Pennsylvania), Katherine Klemperer (Dartmouth College), Richard Marcus (MIT), Joseph Matthews (Inlex, Inc.), Margaret Morrison, (University of Arkansas), Lynne Neufeld, (Telebase Systems, Inc.) and Lawrence Woods (McDonnell Douglas Computer Systems). The intent of the committee was to identify a group of commands to be supported and to define a vocabulary and syntax without specifying the operation of the underlying system. Therefore, what the proposed standard does not contain is perhaps as important as what it does describe:

- o The list of commands that are included in the standard are neither a minimal set nor are they exhaustive. That is, a system may contain more commands and features than are included in the standard. It may also choose not to implement certain commands or features described in the standard. However, to be in compliance, the system must at the very least recognize the command words specified in the standard and display a message indicating that the feature has not been implemented, or some other appropriate message to the user.
- o The standard does not specify the field names or options to be used. Instead, it defines the commands in terms of the types of qualifications and options that ought to be offered. It is then left to the systems designers to decide which specific field names, options, etc. are appropriate in a given application.
- o The standard is not limited to bibliographic data, and in fact, is not limited to text data files, but is intended to be

applicable to all types of machine-readable data.

The goal of establishing a common command language is to create an environment that is as comfortable for the online system user as a rental car. Virtually all makes of cars are slightly different, but they share common features such as the shift pattern that make it possible for anyone to get in a car and drive it away. The same sort of easy transition from one environment to another is the main purpose of a common command language.

A few assumptions guided the Committee in its work to develop the draft standard, Z39.58-198X. These are:

- o Use simple, everyday English words as command names. Use verbs whenever possible.
- o Keep punctuation to a minimum, and where necessary, keep it as simple as possible.
- o Commands may be abbreviated by truncating from the right. All commands have a three-letter unique abbreviation, but an abbreviation of one or two letters is sufficient if it is unique.
- o If a tradeoff is needed, make things easier for the user and harder for the parser.
- o Some details are matters of implementation and not part of the standard. Procedures for correcting mistakes on the screen (is it the backspace key, the rubout, the delete key or something else?); for entering data (is it the Send, Return, or Enter key?); and for paging back and forth in the display (is it the up arrow or the page up key?) are not part of the standard.

The commands are:

EXPLAIN <topic> - provides session-independent explanation of the system, its use, and its database(s)

EXPLAIN commands
EXP databases

HELP - has no options, provides context-specific help.

START - used to initiate a session. Purges the results of a previous session and resets default parameters.

STOP - to end a session. Specific logoff details are system-specific.

CHOOSE <file1> [file1] ... - to select one or more files or databases for searching.

CHOOSE medline

FIND [q1][,q2][...] <term> [[<operator><term>]] [<operator> [q1]...<term>]...

- the basic search command

FIND au Frost
FIND su,ti acid rain and forest\$1

SCAN [q1][term] - used to begin display of a list of terms

SCAN su biology

RELATE [q1][,q2] <term> - displays terms logically related to the specified term

RELATE tranquilizers

DISPLAY [s#][r#][,r#][range][option] - displays records retrieved by the search specified, record(s) specified, and format specified (if any). If nothing is specified, the system defaults will apply.

DISPLAY 5 1,5,10-20 LONG

PRINT [q#][r#][,r#] [range][option] - used to print search results offline.

PRINT 1-10 BRIEF

SORT <sortvalue>[sortvalue2...][s#][r#][,r#][range][order] - used to sort search results according to specified field value(s)

SORT 7 au,ti

MORE [n][,np] - to move forward through the displayed data

MOVE 5

BACK [n][np] - to move backward through the displayed data

BACK 3p

REVIEW [q#][,qn][...] - used to review previously executed search strategies.

REVIEW 1-10

SAVE [Q#] [qname] - used to save search strategies or search results from one session to another.

SAVE S7 mysearch

DELETE <s#>|<ss#>|<m#>|<p#> - used to remove saved search strategies, search result sets, macros, or print requests

DELETE mysearch

SHOW <option> - may be used to display session-dependent information such as default settings, time, news, etc.

SHOW news
SHOW settings

SET <option> - may be used to change session-dependent options such as display format, paging, etc.

SET display long
SET scrolling off

DEFINE <option> <option> - used to rename commands or to set up macros.

DEFINE look display

The proposed common command language standard draft document was distributed for comment during March-May 1986. The comments are being collected by NISO and will be reviewed by the Committee late in 1986.

AUTHOR AFFILIATION: Emily Faven is the Assistant Director for Library Systems, University of Pennsylvania, Philadelphia, PA.

ADDRESS: Van Pelt Library
3420 Walnut Street
Philadelphia, PA 19104-6206

Toward Expert Database Front-Ends

Gabriel Jakobson

Research in artificial intelligence, in particular, natural language processing, knowledge representation, and deductive reasoning, has produced a qualitative change in database user services. Today's database systems use natural language and deductive reasoning, evidence that developers are striving to integrate artificial intelligence and database techniques.

The system described here, IDA (Intelligent Database Assistant), is a new type of database front-end. IDA combines database expertise with an intelligent user interface, giving the user substantial help in query formulation, database selection, and data interpretation. The functional differences between an IDA and a traditional database system include:

- o Querying in a cooperative natural language dialogue mode vs. querying in an isolated command mode.
- o Querying on a conceptual level vs. querying on a factual level.
- o Automatic database selection vs. explicit database specification.
- o Automatic query generation vs. direct specification of data retrieval commands.
- o Portable access to different database systems vs. built-in access to a single database system.
- o Data fusion vs. presentation of separate data chunks.

IDA's structure reflects these expert functions. There are five main components: Communication Monitor, Natural Language Processor, Query Processor, Database Processor, and Domain Specific Knowledge Base.

The success of IDA depends not only on the power of the individual components in the system, but also on the choice of knowledge representation. Because the same knowledge representation (frame) is used throughout the system, from Natural Language Processor to Database Processor, the same set of tools is used to acquire and edit knowledge bases and the same methods are used for clarification and explanation.

IDA is implemented in Interlisp on Xerox 1100 series AI workstations. Currently it is used to access fourth generation relational database management systems, including FOCUS, ADR Dataquery, and SQL.

Artificial Intelligence, Natural Language Processing, Intelligent Database Assistant, Front-End, Intelligent User Interface

AUTHOR AFFILIATION: Gabriel Jakobson is a principal member of the technical staff in the Knowledge-based Systems Department at GTE Laboratories.

ADDRESS: Computer Science Laboratory
 GTE Laboratories, Inc.
 40 Sylvan Road
 Waltham, MA 02254

Thoughts on the Use of Natural Language Interfaces
with Examples from the IRUS/Janus Project

Edward Walker, Ralph M. Weischedel, and Lance A. Ramshaw

The Fleet Command Center Battle Management Program (FCCBMP) offers many substantive, challenging issues for natural language interface research and provides an opportunity to:

- o Demonstrate the results of both past and ongoing research.
- o Transfer natural language interface technology.
- o Perform empirical study regarding language as used in decision-making environments and the effectiveness of heuristics for various linguistic phenomena.

The first version of interface software has been integrated with the FRESH FCCBMP software by Texas Instruments. The Naval Ocean Systems Center is defining the words and semantics for the battle management application.

The paper will emphasize the underlying natural language technology and the system architecture that makes this project possible. It will also highlight the major research areas being addressed and plans for the next generation system.

Natural Language Processing, Fleet Command Center Battle
Management Program, Interfaces

Thoughts on the Use of Natural Language Interfaces with Examples from the IRUS/Janus Project¹

Edward Walker, Ralph M. Weischedel, and Lance A. Ramshaw²

1. Advantages of Natural Language Interfaces

The use of database systems and information retrieval systems is often obstructed by the user's lack of knowledge of the underlying system or its command language. Natural language can be used to provide an interface for such systems that requires little if any training and supports flexible access to a variety of systems by users with widely differing background and intention. Because users already know how to make up commands in natural language, they are not constrained by the artificial syntax of an underlying command language, DBMS or expert system. Assuming they are familiar with the subject matter of the application, they also are largely freed from having to learn or remember non-English file names, field names, and data base codes.

A true natural language interface is far more than a re-implementation of a command language using English-like syntax and vocabulary. Current systems have an extensive model of the application domain and sufficient vocabulary to cover the domain of discourse. They employ sophisticated computational techniques to allow enormous flexibility in the structure of commands.

Natural language interfaces also allow queries to be abbreviated, aggregated, or made more succinct by using such linguistic devices as pronouns, quantifiers, and elliptical constructions. For example, current natural language interfaces would accept as input all of the following ways of asking how many submarines have a combat readiness rating of C3:

What is the number of submarines that are C3?

Count the submarines whose readiness is C3.

Total the C3 submarines.

Give me the number of C3-ready subs.

How many submarines have a readiness that is C3?

Show me the number of submarines with readiness C3.

Tell me how many submarines are C3.

¹The work presented here was supported under DARPA contract #N00014-85-C-0016. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or of the United States Government.

²BBN Laboratories Inc., Cambridge, MA 02238

The breadth of coverage of a system depends on the specific detail and degree of complexity incorporated in its model of the application domain, while its linguistic fluency depends on the computational linguistic techniques it uses to accomplish understanding. Both breadth and fluency are essential to the performance of a natural language interface; however, even the most extensive system cannot cover all the concepts a user might apply to a domain or all the ways of expressing a command, if only because new concepts are bound to arise, and users are bound to invent novel sentence forms or commit unforeseen errors. A good system is one whose coverage closely matches the users' knowledge of the domain and whose fluency allows users to express requests in natural and familiar forms.

2. Examples from a Current Natural Language Interface: IRUS/Janus

A good way to illustrate the abilities of current natural language interface systems is to give examples of how one such system works. The system that will be described here is the IRUS/Janus system, a natural language interface being built by BBN Laboratories and others under sponsorship of the Defense Advance Research Projects Administration (DARPA) Strategic Computing Program.

After 20 years or so of research in computational approaches to natural language, things are now to the point that commercial natural language interfaces to database systems are beginning to appear. It is DARPA's intention to push the state-of-the-art by attempting more challenging and exploratory applications. In particular, the Fleet Command Center Battle Management Program (FCCBMP) is being developed as an integrated system that ties together a database system, a number of expert systems, and a package of various graphical and computational aids. The IRUS/Janus system is being designed as a state of the art natural language interface for this integrated system. An initial version of the interface has already been demonstrated at the Pacific Fleet Command Center in Hawaii.

In addition to its utility as an easy to use interface to a complicated system, the researchers hope that this application of natural language technology will also provide a valuable empirical evaluation of the state of the art and a unique test bed for natural language research and development. The development of the system is expected to result in a very large domain model and lexicon, as well as a significant corpus of user interactions. This is a crucial advantage, as the practical worth of many suggested techniques cannot be determined until they are tested in a realistically large domain with a substantial dictionary and domain model, since the combinatoric difficulties are often the key issue. The problems in natural language processing presented by the complex application domain of the Strategic Computing Program should provide a focus for research activities within the DARPA natural language technology base.

3. The IRUS Natural Language Understanding System

IRUS [1] is the name of the current natural language interface system for the FCCBMP, and is based on the results of years of earlier work at BBN under DARPA sponsorship. Over the course of the next few years, IRUS will gradually evolve into a new system to be called Janus that will incorporate the new and more powerful techniques now being developed.

The overall structure of the IRUS natural language understanding system, is shown in Figure 1 on the following page. As seen there, the translation process between English and the command language of the underlying database or expert system is divided into a number of stages that produce intermediate representations.

In the first stage, the English is translated into a formal semantic representation in an underlying Meaning Representation Language (MRL), a modified form of first order logic. An ATN grammar [9] and parser, along with a lexicon, are used to break the sentence up into meaningful phrases and substructures, and a semantic interpreter converts the phrases into equivalent fragments of MRL and then combines those fragments to get the full interpretation. This semantic interpretation depends on a logical model of the domain which is implemented in the NIKL logical language [4] and on a set of domain interpretation rules attached to the concepts in the model. Parsing and semantic interpretation are interleaved in IRUS, so that the semantics can be used to help guide the parse; for instance, the attachment of the prepositional phrases in the following two sentences depends on their semantics:

She fed the cat with the white paws.

She fed the cat with the eye dropper.

A separate process then resolves any anaphora or ellipsis found in the initial MRL.

Translating from the MRL into the command language of the underlying system is itself a two-stage process. The MRL forms are first mapped into an Extended Relational Language (ERL) [6, 7], which reflects the logical structure of the underlying system. For instance, rather than an MRL logical variable of type "VESSEL", ERL refers to tuples in the VESSELS table of a logical relational database. From the abstract ERL version of the query, a code generator creates the actual commands in the language of the target system. Here is where details like the actual file names and field names of the underlying system are supplied. Both phases of this translation depend on their own sets of translation rules.

4. Domain Independence and Modularity in IRUS

One of the key questions with a system like IRUS is how closely tied it is to the domain and underlying systems for which it was originally developed. The system must be carefully designed for modularity if it is to be possible to make use of it later in different domains or with different underlying systems. In IRUS, the conceptual view of the domain and the operation of the underlying natural language processing system are separate from each other, just as the details of the particular database system or expert system being used are separate from the natural language processing system itself. Thus parsing and semantic interpretation are expressed in terms of the users' conceptual organization of the application, so that the underlying system can be changed with minimal repercussions in the natural language system. Similarly, the IRUS knowledge bases which contain specific information about the application domain may be filled or modified over time to extend the breadth and fluency of the basic system without making software changes to the underlying linguistic processing modules.

The major domain independent modules of the IRUS system include the parser and associated grammar, the semantic interpreter, and the subsystem for resolving

STRUCTURE OF IRUS

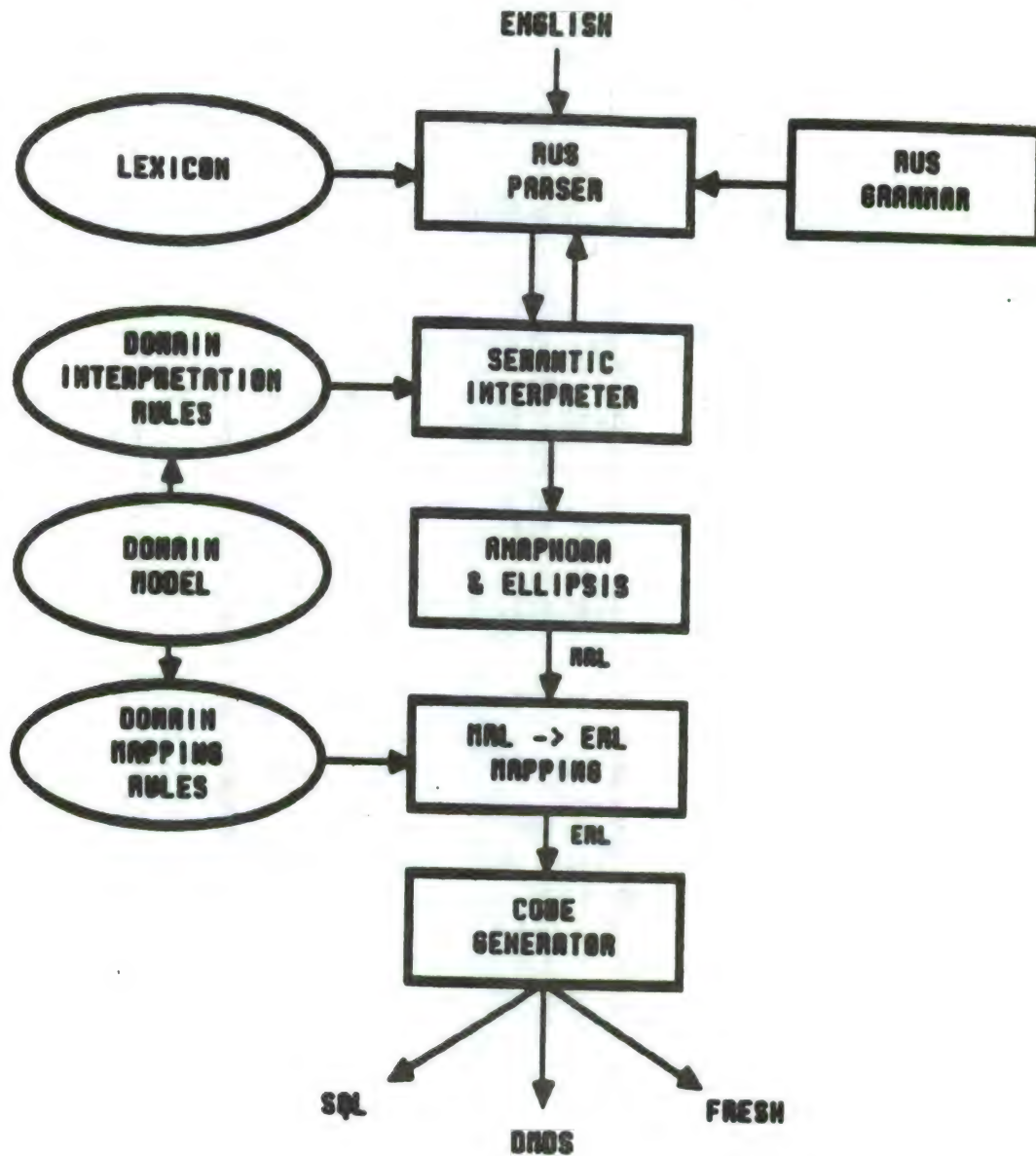


Figure 1: Structure of the IRUS Natural Language Interface Program

anaphora and ellipsis. The system could be brought up in a new domain without any significant changes to those modules by simply supplying the appropriate new lexicon, domain model, and set of semantic interpretation rules. In the same way, the code for translating from MRL into the target system command language is largely independent of the application, though it makes use of an application-specific set of translation rules.

Supporting tools have been designed and implemented [5] to aid developers in the tasks of building a domain model and of constructing the domain specific dictionary and semantic interpretation rules, so that the task of bringing up the system in a new domain can be assigned to personnel who already understand the domain, even if they are not highly trained in computational linguistics.

5. Enhancements Suggested by the Application

Bringing up IRUS in the Navy domain as part of the Strategic Computing program has already begun to provide "technology pull", i.e., to raise concrete problems that arise naturally in realistic applications in order to motivate and focus the progress of the technology. One example is reasoning about time, with a history of past events and a schedule of future events.

While an adequate semantic representation for time is still an open research issue, as exemplified by alternative approaches representing time as points or as intervals, some way needs to be found to provide adequate reasoning power in support of time references. Consider the differing, implicit dates needed to answer the following pair of questions:

Where was the Kennedy Tuesday?

Where will the Kennedy be Tuesday?

Special notations peculiar to the application domain also must be explicitly added to the grammar of IRUS. Examples include latitude-longitude positional notation and Greenwich Mean Time.

A major piece of "technology pull" in the FCCBMP domain comes from the presence of multiple underlying target systems of different types--a database system, an expert decision support system, and a display/calculation command language. Thus the mapping rules and code generation capabilities of IRUS must be expanded to be able to distinguish between the different underlying systems and to produce the appropriate code for each.

It is also anticipated that users will want to issue stock sequences of commands or queries and employ not only a keyboard, but also a pointing device to generate input, so provision is being made for insertion of strings generated by selecting a graphic object into a text query.

6. Janus: The Evolution of Natural Language Technology

The IRUS system described in the previous sections represents the state of the art in natural language technology. However, significant advances should be possible over the course of the Strategic Computing Program; therefore, an evolutionary sequence of systems, called Janus, will be developed over the remaining course of the project to embody these advances.

At the most abstract level, we may view a natural language interface as consisting of four software components. An understanding component must take English input from the user and provide a formal semantic representation for the meaning of that input.

The component that receives that formal meaning representation must map the user's request into the detailed information structures and query syntax of the actual underlying data base or expert system. Furthermore it is the responsibility of the second component to identify from the user's request which underlying system(s) must be accessed to fulfill the request. In the long run it is highly desirable that the selection of a system be invisible to the user.

Once the underlying system has fulfilled the request, a third component must select an appropriate medium for the answer, such as graphical output, textual information, or tables. In addition, some extended response may be appropriate, particularly if the request is not directly fulfillable. Given a request for which English output is appropriate as part of the answer, then transforming that part of the answer into readable, unambiguous English is the responsibility of the fourth component, a natural language generation system.

IRUS incorporates only understanding of natural language; there are no sophisticated English output capabilities. The Janus effort will include a significant attempt to build a complete system including both understanding and generation capabilities, with the integrated system containing work from the DARPA natural language research community.

In broad terms, the research planned for Janus falls into several categories:

- o parallel processing algorithms: While the speed of the understanding process itself is not a problem, there are some circumstances that require the system to explore many alternatives at considerable depth before it can select the desired interpretation. Handling ill-formed input like "List the subs and there readiness" is one such case; interpreting speech input is another. The depth provided by parallelism will be essential to provide robust handling of such phenomena.
- o models of discourse and models of the user: Such models are crucial for the system to be able to correctly resolve pronouns, other references, and ellipsis, as in "List the ships in the Indian Ocean. <answer> Carriers?"
- o modeling the intent of the user: A purely literal interpretation can miss the intent of the user; a simple example is "I can't see the Enterprise," where this must be interpreted as a request for help to locate it. A model of user intention can also provide crucial information for interpreting ill-formed input.

- o richer knowledge representation mechanisms: The logical language has to be made richer in order to have greater coverage of the semantics of English, including, for example, generics like "C3 ships should not be deployed" and propositional attitudes like "They believe that the Frederick downgraded to C4".
- o more sophisticated models of the reasoning underlying system capabilities and user intentions: These are necessary in order to route queries to the appropriate underlying system and construct appropriate responses; for instance, "Where are the carriers?" might be appropriately answered by a graphical display, a table, or English text, depending on the context.
- o more sophisticated plans for generating readable text, including multiple sentence texts, paragraphs, and, eventually, short reports: The generation system needs to be able to create clear but concise descriptions of domain entities, telling the user enough to recognize the object without saying everything it knows, and to organize its output in an understandable way. There is ongoing research at USC/Information Sciences Institute on components for text planning and generation [3].
- o new classes of helpful responses: Direct answers can often be helpfully supplemented with information derivable from the domain model. Work in this area is proceeding at the University of Pennsylvania [8, 2]

7. Summary

Natural language interfaces can be very useful and even essential parts of information retrieval systems, since they can insulate the new or casual user from the complexity of the exact structures and command languages of the multiple underlying systems available to the user. Well-designed, modern natural language interfaces can already provide far more power and flexibility than comes from just an English-like command syntax. This paper has tried to provide an overview of how such interfaces work and of the issues that are now being explored to extend their breadth of coverage and their fluency.

AUTHOR AFFILIATION: Ralph Weischedel is a Senior Scientist at
BBN Laboratories, Cambridge, MA.

ADDRESS: BBN Laboratories
10 Moulton Street
Cambridge, MA 02238

References

- [1] Bates, M., Stallard, D., and Moser, M.
The IRUS Transportable Natural Language Database Interface.
Expert Database Systems.
Cummings Publishing Company, Menlo Park, CA, 1985.
- [2] Finin, T., Joshi, A.K., & Webber, B.L.
Natural Language Interactions with Artificial Experts.
Technical Report MS-CIS-86-16, University of Pennsylvania, 1986.
- [3] Mann, W.C. and Matthiessen, C.M.I.M.
Nigel: A Systemic Grammar for Text Generation.
Systemic Perspectives on Discourses: Selected Theoretical Papers from the 9th International Systemic Workshop.
Ablex, Norwood, NJ, forthcoming.
- [4] Moser, M.G.
An Overview of NIKL, the New Implementation of KL-ONE.
In Sidner, C. L., et al. (editors), *Research in Knowledge Representation for Natural Language Understanding - Annual Report, 1 September 1982 - 31 August 1983*, pages 7-26. BBN Laboratories Report No. 5421, 1983.
- [5] Moser, M.G.
Domain Dependent Semantic Acquisition.
In *The First Conference on Artificial Intelligence Applications*, pages 13-18.
IEEE Computer Society, December, 1984.
- [6] Stallard, D.
Data Modelling for Natural Language Access.
In *The First Conference on Artificial Intelligence Applications*, pages 19-24.
IEEE Computer Society, December, 1984.
- [7] Stallard, D.G.
A Terminological Simplification Transformation for Natural Language Question-Answering Systems.
In *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, June, 1986.
- [8] Webber, B.L.
Question, Answer and Responses: Interacting with Knowledge Base Systems.
Technical Report MS-CIS-85-50, University of Pennsylvania, 1985.
- [9] Woods, W.A.
Transition Network Grammars for Natural Language Analysis.
CACM 13(10):591-606, October, 1970.

A NATURAL LANGUAGE FRONT-END FOR DATABASE UPDATE*

Sharon Salveter

Natural language database access requires support of both query and update capabilities. Although a great deal of research effort has been expended to support natural language database query, little effort has gone to support update. We describe a model of action that supports natural language database update, and the implementation of a system that supports the model. A major goal of this research is to design a system that is easily transportable to both different databases and different DBMSs.

Artificial Intelligence, Natural Language Processing, Front-End

*This research is partially supported by NSF grants IST-8214622 and IST-8408551. A similar version of this paper was published in IEEE Database Engineering, December 1984.

1. Introduction

Database access includes both query and update. In order to access a database, an end-user has traditionally had two options. He could learn the database structure and the DML required by a particular database, and formulate the access request himself. Alternatively, he could explain his request to a programmer who then writes the DML. Both options have serious drawbacks. In the first, it is not always reasonable for a possibly naive user to learn a formal DML and database navigation strategies. In addition, because of database integrity constraints and view update problems, users are often prohibited from writing transactions that update the database. The second option places a level of administration between the user and the database that is generally cumbersome, and which is inappropriate for a user sitting at a terminal in his office. Such an approach is particularly inappropriate for personal databases.

In order to avoid these impediments to database access, it is desirable to support natural language database access. Although a great deal of research effort has been expended in support of natural language database query [HARR77, DAME78, WOOD76, KAPL79, WALK78, WALT78], and at least one commercial system is available [HARR79], little effort has been expended in support of natural language database update, as noted by Wiederhold [WIED81]. In this paper, we describe a system that supports natural language database updates that are admissible for a given database. An additional goal is to design a system that is easily transportable both to different databases and DBMSs.

2. The Update Problem

Previous research [SALV82] has shown that it is not possible, in a natural manner, to extend natural language query systems to support update. In order to support natural language database query, a computer system must be able to represent a *stative correspondence* between database states and real world states, as shown in Figure 1. This stative correspondence logically connects database objects with real world entities and relationships.

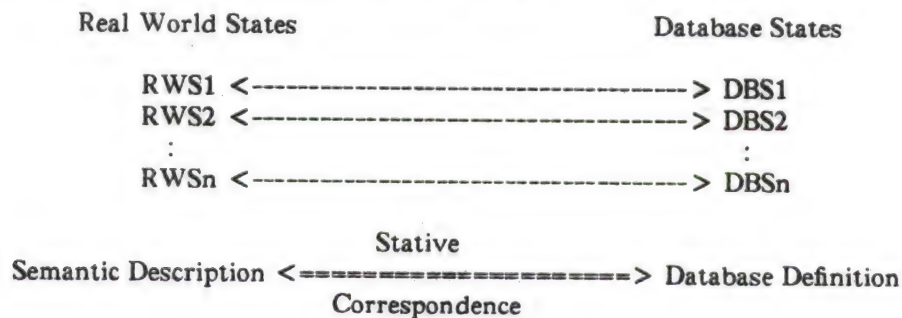


Figure 1
Real World State - Database State Connection

The stative correspondence is not adequate for supporting natural language database update. In Figure 2, we see that when some action in the real world causes a state change from RWS1 to RWS2, we must execute a DML sequence to change the database state from DBS1 to DBS2. Given an update command that describes a real world action we need to find a DML sequence that will accomplish the corresponding change in the database. We need to specify *what* is to be modified and *how* to make the modification. We need to connect active verbs, such as "hire" and "schedule" with structures that dictate DML sequences that perform the corresponding database update. We need to represent an *active correspondence* between natural language descriptions of actions and DML sequences.

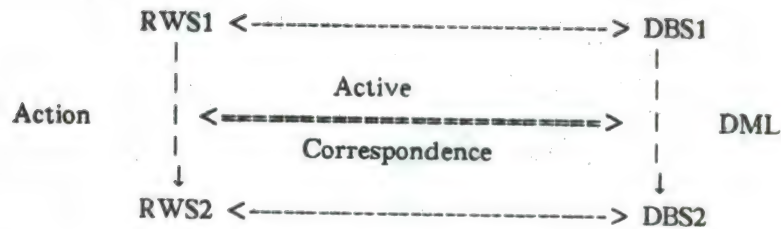


Figure 2
Real World Action - DML Sequence Connection

3. System Overview

Our research has resulted in a formal language for specifying what real world actions mean *with respect to* a particular database, and a system that executes it. The formal structure that links natural language update commands to DML sequences is called a *verbgraph*. The verbgraphs are designed in tandem with the database. During the database design process, the design staff analyzes a real world enterprise and creates a database scheme that is an appropriate formal model of that enterprise. A critical consideration in designing a good database scheme is how the database will be *used*. That is, what types of queries will be addressed to the database, what kinds of update transactions will need to be performed, and what consistency constraints must be satisfied. Our approach provides the design staff with a formal language for specifying update transactions*, just as the DDL of a DBMS is a formal language for specifying database schema. The formal representation can then be processed by our system, just as a DBMS interprets the DDL to create and maintain a database instance. Our system facilitates formal and systematic capture of domain-specific database knowledge, which heretofore resided only in a programmer's head.

The overall architecture of our transportable system is shown in Figure 3.

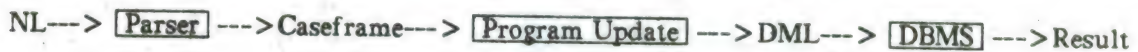


Figure 3
Domain-Independent Natural Language Database Access Architecture

The parser embodies domain-independent general linguistic knowledge. From the natural language update command it produces a caseframe representation [CHAR76]. Briefly, a caseframe is a collection of named slots, such as AGENT or OBJECT, which can be instantiated. We are currently using simulated output of the RUS parser [BOBR78]. Program Update embodies domain-dependent knowledge. It takes the formal caseframe representation as input, links linguistic elements to database objects and updates, and produces a DML sequence that represents what the natural language command means *with respect to* this database. We are using the INGRES DBMS [HELD75]. Actually, we are not tied to a particular DBMS, as Figure 3 implies. Our system produces a representation in a formal intermediate language (IL) that can be translated into the DML of a given DBMS. Thus, the overall architecture looks like Figure 4.



Figure 4
Transportable Natural Language Database Access Architecture

An IL-to-QUEL translator has already been written [ASSI84]. Thus our attention is limited to the

* Although we have not yet implemented query capability, there is no theoretical impediment.

nature of the caseframe-to-IL conversion. The architecture of this component is shown in Figure 5.



Figure 5
Caseframe-to-Intermediate Language Translator Architecture

Because the IL-to-DML translation is a well-understood problem, and not of further interest here, we sometimes speak of the IL as the language that actually updates the database. The verbgraphs are a set of structures that represents what various natural language update commands mean with respect to this database; they must be specified for each database. The control program directs execution of the verbgraphs; it need never be rewritten. This design is analogous to expert systems where domain-dependent information is represented in production rules. There, the control program is domain-independent; it controls selection and execution of productions.

4. The Verbgraph

A natural language verb may have a number of different senses. A *sense* of a verb roughly corresponds to the different definitions that might be given in a dictionary. For example "run" has at least the three senses: a person moving quickly, a machine operating, and a person campaigning for public office. In our scheme, a verbgraph represents a single verb sense. Thus one verbgraph might represent the "hiring" action, regardless of how the update is specified in natural language. A given verb sense may have a number of *variants*. For example, hiring faculty may require different database actions than hiring secretaries. A verbgraph represents all legal variants of an action, and is also used to determine which variant is specified in the natural language command. The set of verbgraphs for a database is the repository for several kinds of information (constraints and default values may eventually be supported by some ideal DBMS):

1. Linkage of linguistic constructs to database objects and updates. (Use the AGENT case as the value of attribute NAME in EMP, "hiring" results in insertion into EMP.)
2. Constraints on the database. (Maximum of 40 students in a course.)
3. Default values. (All courses are 4 credits, unless otherwise specified.)
4. Parameterized IL update commands that will ultimately comprise the update transaction.
5. Database retrievals that may have to be performed in order to process the update request.
6. Questions to ask the user if insufficient information is specified in the natural language command.
7. Templates for tuples to be inserted into, deleted from, or modified in the database.

A verbgraph is composed of a tree and a blackboard, as shown in Figure 6a. The tree controls instantiation of the blackboard, which contains objects that will ultimately determine the IL sequence. The tree has at most two levels*. There is a distinguished root node and an arbitrary number of leaf nodes. Each node contains a *guard*, a boolean expression. The root node has guard true. A variant is defined by the accumulation of all the nodes in the tree whose guards evaluate to true for a given input. The order of execution of the leaf nodes is unimportant. When the

* A two-level tree simplifies our control strategy. We could allow N-level trees, as discussed in [SALV84].

may be determined but the required data incomplete. The questions component of the blackboard is a place for specifying canned natural language questions to be asked of the user.

The Tree

The blackboard stores information needed by the tree. However, the control program does not directly access the blackboard: it selects a set of nodes and executes them. The nodes are responsible for ensuring that the correct IL transaction is constructed. It is in the tree, then, that linguistic objects are linked to database objects, real world actions are linked to database updates, database integrity constraints are specified, and the blackboard objects are manipulated. Conceptually, the root of the tree represents what is true for all variants of this verb sense, and each leaf represents that which is the case for some aspect of a variant. (Recall that a variant is defined by the root and all leaves, or aspects, whose guards evaluate to true.) The tree is specified by a series of node definitions, as shown in Figure 6a. One node is the distinguished root, the remaining nodes are the leaves. The format of a node is shown in Figure 6b. A node consists of three parts: guard, prerequisite, and action.

The *guard* determines whether a node is selected for execution. Guards test caseframe values. Because a variant is determined by information in the natural language sentence (and therefore the caseframe), the guard can only compare caseframe values against constants, caseframe values, and results of retrievals on the database. The *prerequisite* component indicates those caseframe slots that must have values before node execution is to proceed. If any specified caseframe slot does not have a value, a blackboard question is asked of the user, and processing is suspended until the caseframe slot is instantiated.

Actions are the crux of the node. They perform a wide variety of functions: instantiate blackboard variables, ask the user for more information, "check off" blackboard IL, retrieve database values, specify default values, and specify database integrity constraints. An action is either a checkoff or an assignment. The various forms are best illustrated by example. A checkoff is the simplest action. The statement check (A3) causes the blackboard IL sequence labeled A3 to be checked off on the blackboard; it will be part of the final IL transaction.

The basic function of an assignment is to instantiate a blackboard variable. The value may be:

- a constant: `t.X := 5`
- a system variable: `t.X := clocktime`
- a blackboard variable: `t1.X := t2.Y`
- a caseframe value: `t.X := cf//location`
- a default value: `t.X := cf//duration | default(1hour)`

If the caseframe slot named duration has no value, use 1 hour.

- the result of a database retrieval: `t.X := R1`

R1 is the label of a blackboard retrieval that returns a single value.

- required to be in a range of values: `t.X := cf//location in R1 # Q8`

R1 is the label of a blackboard retrieval, Q8 of a question. If the `cf//location` value is in the result of R1, then use it. Otherwise, ask question Q8 and wait for a response.

- required to be in a range of values if it exists, otherwise a default is used:

`t.X := cf//location in R1 # Q8 | default(R2)`

If `cf//location` has a value and it is in R1, then use it. If it has a value but it is not in R1, ask Q8 and wait for a response.

If `cf//location` does not have a value, use the default, which is the result of executing blackboard retrieval labeled R2.

A formal description of the verbgraph language, a more complete description of the system, and illustrative examples can be found in [SALV84].

5. Concluding Remarks

A prototype of this system is implemented in C on a VAX 11/780 running Berkeley UNIX.

We plan to implement a toolbox for verbgraph specification, which will include a structured editor. We will also provide an interactive verbgraph debugger, which will allow the verbgraph designer to test the correctness of the verbgraphs. He will be able to test which verbgraph nodes evaluate to true for a given input or class of inputs, step through instantiation of blackboard variables and IL checkoff, and query the consistency of verbgraph components.

We also need to address the complex problem of run-time interaction with the user. For example, the user may supply additional information that changes which node's guards evaluate to true. We propose to design our end-user interaction package so that minimum reprocessing is necessary, while allowing maximum flexibility. An important design criterion is to avoid forcing the user to restate information that has been given previously.

Acknowledgments

Douglas Stumberger was intimately involved in the design of the verbgraphs and system architecture, and is responsible for a large part of the implementation. Michael Siegel participated in the verbgraph design. Thomas Schutz assisted in the implementation.

References

- [ASSI84] Assiff, S. Intermediate Language to QUEL Translation. BU CS Dept Masters Thesis. 1984.
- [BOBR78] Bobrow, R. The RUS System. BBN Report #3878, 1978.
- [CHAR76] Charniak, E. and Y. Wilks, *Computational Semantics*, North Holland, 1976.
- [DAME78] Damereau, F., The Derivation of Answers from Logical Forms in a Question Answering System. *American Journal of Computational Linguistics*, microfiche 75, 1978, pp. 3-42.
- [HARR77] Harris, L., Using the Database itself as a Semantic Component to Aid the Parsing of Natural Language Database Queries. Dartmouth College Mathematics Department TR 77-2, 1977.
- [HARR79] Harris, L., Experience with ROBOT in 12 Commercial Natural Language Database Query Applications. *Proceedings of the International Joint Conference on Artificial Intelligence*, Tokyo, 1979, pp.365-368.
- [HELD75] Held, G., M. Stonebraker and E. Wong, INGRES - A Relational Database Management System. *Proceedings of the 1975 National Computer Conference*, 1975.
- [KAPL79] Kaplan, S.J., Cooperative Responses from a Natural Language Database Query System. Stanford University TR HPP-79-19, 1979.
- [SALV82] Salveter, S. and D. Maier, Natural Language Database Updates. *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics*, Toronto, 1982, pp.67-73.
- [SALV84] Salveter, S., Natural Language Database Update. BU CS Dept TR# 84/001.
- [WALK78] Walker, D., *Understanding Spoken Language*. American Elsevier, 1978.
- [WALT75] Waltz, D., Natural Language Access to a Large Database: An Engineering Approach. *Proceedings of the International Joint Conference on Artificial Intelligence*, 1975.
- [WIED81] Wiederhold, G., S.J. Kaplan and D. Sagalowicz, Research in Knowledge Base Management Systems. *SIGMOD Record*, 7, 3, April 1981, pp.26-54.
- [WOOD76] Woods, W. et al, Speech Understanding Systems: Final Technical Report. BBN Report #3438, 1976.

AUTHOR AFFILIATION: Sharon Salveter is an Associate Professor of
Computer Science at Boston University.

ADDRESS: Computer Science Dept.
Boston University
111 Cummington Street
Boston, MA 02215

USING MENU-BASED NATURAL LANGUAGE TO QUERY AN INTEGRATED DATABASE MANAGEMENT AND INFORMATION RETRIEVAL SYSTEM

Craig Thompson and Steve Martin

Using abstract data types and indices, information retrieval query and indexing capabilities are added to a lisp machine-based relational database management system, unifying these two closely related sorts of information management systems. A menu-based natural language interface generator is augmented to provide uniform access to both sorts of systems.

Relational Databases, Information Retrieval, Abstract Data Types,
KWIC Indices, Natural Language Interfaces, Menus

1. INTRODUCTION

Database management systems are designed to handle formatted record data. Document retrieval systems are designed to handle unformatted textual data, providing responses to queries where the identifying vocabulary is large and open-ended. Both sorts of information management systems are similar. Both can be accessed by query languages (attribute qualifiers vs boolean combinations of keywords, respectively) and both are routinely implemented with fairly well understood standard index types (tree, hash, or heap vs KWIC or variants, respectively). A combined database and information retrieval system would be useful, but, unfortunately, database management systems and information management systems are not well integrated.

Many information retrieval systems (eg DIALOG [28]) provide specific language constructions to access specific fields (eg AU for author in DIALOG) and do not allow a user to define new structured object types or to manipulate (insert, delete, ...) stored records. In relational systems, end-users can specify table structures. Also, in relational systems, indices can be created or destroyed without affecting logical properties of retrieval. But, index structures in information retrieval systems are usually built in. Relational database management systems, on the other hand, provide only limited capabilities to deal with textual data records, a class of data that includes not only natural language texts but also programs and other unprocessed data. For instance, there seem to be no relational systems that support mixed relational-style and boolean-style queries, supporting variable length text strings and also supporting both database and information retrieval index types.

The goal of this paper is to present such an integrated system. The key to our approach is to make use of a facility in the Explorer Lisp Machine Relational Table Management System (RTMS) to allow the end-user to define new data types and to support them with corresponding user-specified indices. In addition, the paper describes extensions to a natural language interface that allow an end user to ask mixed DBMS and IR queries in a limited dialect of natural language. The resulting implemented system allows a user to define a COURSE relation with KWIC-indexed fields TITLE and DESCRIPTION and balanced tree-indexed fields DEPT and COURSE# and to issue queries asking to find instructors in CS, psychology or English that teach courses that involve database, information retrieval, text analysis, or AI.

The remainder of this paper is organized as follows. Section 2 describes some past approaches to wedding database and information retrieval systems. It then describes RTMS, concentrating on the facilities for defining abstract data types in RTMS. It ends by describing specifically how support for information retrieval was implemented in RTMS. Section 3 describes menu-based natural language interfaces and then discusses extensions to an interface generator that facilitate

building menu-based natural language interfaces to a DBMS augmented with information retrieval capabilities.

2. AUGMENTING A RELATIONAL DBMS FOR INFORMATION RETRIEVAL

2.1 PAST WORK

Relatively little work has focussed on combining database management and information retrieval systems.

Haskin and Lorie [12] discuss extending SQL to support long fields for storing non-coded data, but only explore operators on fields like fetch, insert and update. Stonebraker et al [20] propose to enhance a relational dbms to support document processing. They suggest extensions to INGRES to include variable length strings, ordered relations, and new substring search and concatenation operators. They are mainly interested in using relations to represent text in various arrangements (eg one line per tuple) to support implementations of text editors and formatters.

Macleod [14] discusses how a macro facility can be built on top of SEQUEL to translate IR-like queries to SEQUEL queries. Crawford and Macleod [4] show how IR indexing tools including negative dictionaries, stem dictionaries, and thesauri can be stored as relations and manipulated using relational operators. Crawford [5] proposes a set of relations to explicitly index text and shows how a bibliographic database can be represented in fourth normal form in a relational database and how several common IR queries can be performed within the relational database model in a variety of query languages. King et al [10] point out that normalizing document entries may lead to fragmenting complex objects across several relations, increased storage utilization, and increased page faults unless clustering techniques are available. They suggest a specialized hashing scheme for their application and do not suggest extending it for a general purpose dbms. Schek and Pistor [17] begin with motivations similar to those of this work, to provide support for information retrieval, but diverge by providing a general scheme for supporting non-1NF structured information in a relational database based on nest and unnest operators to build and unpack structured data items.

Kolodner [11] and Salton [15] suggest uniform representation formalisms for text, tables, and logical assertions based on conceptual dependency and composite concept vectors (respectively), which are both similar to predicate argument representations. The problem with these approaches is that practical means of translating text to these conceptual representations in a domain independent way are not yet well understood.

These works complement the current work in suggesting other techniques for extending a relational database to allow queries of textual fields of records, but fail to suggest that such queries can be directly supported by indices tailored for IR.

2.2 ABSTRACT DATA TYPES AND INDICES IN RTMS

Stonebraker et al [19] discusses the need for supporting abstract data types and indices in a relational database, illustrating the ideas with examples from a CAD domain. This section describes RTMS, a relational table management system for lisp machines, concentrating on RTMS's parallel capabilities for handling abstract data types.

Functionally, RTMS implements most standard relational database capabilities. Data definition functions DEFINE, RENAME, or DESTROY databases, relations, views, attributes or secondary indices. There is no restriction on relation degree or cardinality. A relational algebra includes operators RETRIEVE (select and project), JOIN, UNION, DIFFERENCE, and INTERSECTION as well as operators INSERT, DELETE, MODIFY, GRANT, and REVOKE. Conventional databases (eg INGRES [18], SQL [1]) provide the same sort of functionality as described for RTMS, but in an environment where the only primitive data types are fixed length character strings, numbers, and dates, where the only functionality available is contained in the data manipulation language, and where interfaces to host languages are awkward to use.

In a lisp environment, users would find a relational database less useful if it failed to provide support for lisp data objects. Hence, RTMS allows users to use any domains they want to define. Users can define domain dependent functions for validating the data type of domain elements inserted into relations, for domain-dependent partial ordering functions (eg GREATER-THAN-IN-LENGTH), and for flattening and restructuring complex structures so that they can be saved to disk. Default domains are also supported to allow users to ignore the details of domain definition in simpler applications. Any type of data object may be stored in a relation including pointers, lists, flavor instances, arrays, variables, file names, relation names, graphical objects and large amounts of text. In the pure relational model, components of relations are atomic, but RTMS recognizes that atomicity is application dependent and does not enforce artificial restrictions requiring flattening complex data objects.

The lisp environment is a one language environment. The relational algebra of RTMS is implemented with lisp functions so it is easy to embed in lisp programs. The ability to embed database calls in programs in a one language environment makes passing parameters between the database operators and other lisp functions straightforward, requiring no specialized mechanism. Within RTMS, names of databases, relations, and attributes follow the same rules as do names of lisp variables. In addition, it is natural to allow lisp predicates in the WHERE clause of RETRIEVE statements and in the predicate in theta JOINS. For instance, if a user defines a predicate KEEP-TUPLE which pops up a display of a tuple and allows him to mouse KEEP TUPLE or REJECT TUPLE menu choices, then he can include KEEP-TUPLE at the end of a WHERE

clause s-expression, allowing him to select or reject most tuples automatically based on the first part of the WHERE clause and manually select from the remaining tuples as in:

```
(retrieve from 'EMPLOYEES into 'EMPLOYEES-OF-INTEREST
  where (AND (GT age 30) (OR (GT salary 50K) (KEEP-TUPLEP))))
```

In another example, a user wants to "Find the interstate highways in Texas that go through Dallas". To do this, he might join the relations HIGHWAY(name, type, graphics-subpicture, ...) and CITY(name, population, graphics-subpicture) using an application-dependent join predicate GRAPHICS-INTERSECTP which returns true if two graphics polyline entities intersect.

In RTMS, tuple structures and table storage structures can be user-defined using accessor functions. RTMS comes with a predefined collection of useful tuple implementations including tuples implemented as lisp lists, structures, and flavor instances. In addition, RTMS provides a predefined collection of storage structures including balanced tree, hash, and heap useful for building indices. A user can specify the tuple implementation and storage structure when defining a relation. In addition, RTMS provides an operation called ATTACH-RELATION, which allows a user-defined data structure that matches one of the tuple implementation-storage structure implementations to be grafted into RTMS as a database relation without recopying, thus allowing a user easy access to RTMS query functionality, including the ability to build other indices on the data structure to provide alternative access paths. A corresponding DETACH-RELATION operation allows a user to remove a database structure from the database without destroying it. An application of these capabilities is to facilitate the building of a RELATION-MIXIN which adds to a lisp flavor the ability to manage and access instances of the flavor and perform set operations on them.

A user with other favorite storage structures than those presently provided by RTMS can define them to RTMS by defining a set of accessor functions. These functions include RETRIEVE-<IMP>-<STO>, INSERT-<IMP>-<STO>, and several others. These functions serve to isolate the implementation level of RTMS from the functionality the user sees. Not much work is involved in defining these accessors since their basic job is to provide a means of registering a foreign data structure's database equivalent operations for traversing and modifying the structure.

The added flexibility offered by RTMS must be accompanied by a concern for efficiency. Optimization problems can occur when users are given ready access to lisp. In operators like RETRIEVE and JOIN, allowing user-defined functions can hide from RTMS the opportunity to use defined indices making it necessary to loop through all tuples in a relation executing the user-defined function. For instance, in the query

```
(retrieve from 'employee where (equ (string_length name) 10.)),
```


if the form (EQU (STRING LENGTH var) const) is not registered with RTMS, the form must be executed for each tuple in R. When the user is working with personal sized databases or when he has a new need unsupported by current RTMS functionality, this solution is acceptable. But a framework for admitting optimization hints to RTMS can allow the system to retain the ability to optimize while allowing extra flexibility. The way RTMS handles these hints is to provide an optimization handler for user-defined functions (forms), so that AND, OR, EQ, GT, LT, and other commonly used functions are supported. To plan how the RETRIEVE will be implemented, RTMS interprets the WHERE clause as long as it is not blocked by user-defined functions. Thus, in the WHERE clause:

(AND (LT age 40) (OR (GT age 30) (KEEP-TUPLEP)))) ,

if an ordered index on AGE exists, only tuples corresponding to employees under 40 years old need be considered, all tuples between 30 and 40 must be returned, and KEEP-TUPLEP must be executed on all remaining tuples. Users must beware of including side-effects in user-defined functions, since the availability of indices can affect which tuples user-defined functions are executed on. The optimization capability is extensible; users can register a definition for OPT-<sto>-<opt-fn> with RTMS to inform RTMS of the opportunity to use a supported storage structure in implementing a new function, as when one defines OPT-TREE-STRING LENGTH to tell RTMS to use a TREE structured index ordered by string length to locate strings of length 10 in the example above.

In sum, the interesting contribution of RTMS is to make the functionality of a relational database more easily extendable by the user and to be well integrated with the lisp environments. Often increased functionality is paid for by decreased efficiency. Proponents of the conventional relational model claim that the physical tree structures of a hierarchical data model can be implemented at the storage level while still giving users a logically independent tabular view of the data. To take full advantage of the hierarchical storage, a user would have to restrict himself to just the physically supported join or index paths. Yet, the extra flexibility of requesting logically meaningful searches over unsupported paths is the central strength of the relational model. In RTMS, the ability to add user-defined domains, search predicates and indices makes RTMS more useful in a lisp environment. A user seeking efficiency can restrict himself to supported data types, search predicates, and access paths. A user seeking flexibility can more easily drop into lisp to build more complicated search strategies. If those search strategies are important for his application, he has a route to add a more efficient implementation to RTMS. We expect to be able to use these hooks to define new implementation types that remove the restriction that relations must be in core, without affecting RTMS.

2.3 ADDING INFORMATION RETRIEVAL CAPABILITIES TO RTMS

The implementation of abstract data types in RTMS made adding IR queries straightforward, and, in fact, the entire implementation effort described in this paper took only about 40 man hours for a lisp-machine proficient new RTMS user with menu-based natural language background. The first step was to implement a lisp predicate (independent of RTMS):

```
(MATCH boolean_query string1 ... stringN) --> boolean
```

MATCH returns true if the boolean_query matches any of the remaining string arguments. The boolean query may contain OR, AND, NOT (with that precedence) or terms which may be words or strings containing wildcards like * and ? or adjacency patterns like word1(nW)word2, where word1 and word2 appear in order with n or fewer intervening words. The boolean query may be specified either as a string using infix or as a list using normal lisp reverse polish.

This extension was enough to immediately add IR query capabilities to RTMS so that a user could define a relation as:

```
(defrelation course
  (department number title description textbooks units))
```

and could insert tuples like:

```
(insert 'course
  :department CS
  :number 4750
  :title "Interactive Problem Solving Tools"
  :textbooks ("Abelson and Sussman, Structure and
    Interpretation of Computer Programs,
    The MIT Press, 1985")
  :description "Principles of symbolic programming and problem
    solving. Assignments involve writing pieces of
    interactive tools. Tools are collected into a
    library that grows as the course progresses.")
```

and could then use mixed DBMS and IR qualifications in querying this relation as in:

```
(Retrieve from course
  where (and (eq department CS)
    (MATCH "comput* and
      (linguistics or natural(w)language)"
      ,title ,description ,@textbooks)))
```

to find courses in the CS department that involve computational linguistics.

As defined, the lisp predicate MATCH functionally adds IR capabilities to RTMS. The next step was to define a KWIC index capability for RTMS so that users could build secondary IR

indexes to support IR queries more efficiently. The goal was to be able to issue statements of the form:

(defindex INDEX-NAME on ATTRIBUTES of TABLE with type = kwic)

Executing a statement of this form has the effect of building a hash structured index that associates a list of <record, field, offset> triples with each distinct word appearing in the "attributes" fields of tuples in the "table". To implement this KWIC index storage structure, routines had to be written to create, manipulate (by insertion, deletion, or modification), or destroy such indexes. In addition, a new implementation of MATCH was needed to take advantage of an index so that OR, AND and NOT could be evaluated by set union, set intersection, and set difference operations on index sets. Finally, the query optimization component of RTMS had to be informed of when to use the IR index if it were present, by defining function OPT-KWIC-HASH. Because RTMS already provided HASH indexes (on whole fields) and KWIC indexes were implemented using hashing on words in fields, operations on KWIC indexes were very similar to operations on HASH indexes and only small modifications to the existing code were needed to complete the implementation.

3. PROVIDING A MENU-BASED NATURAL LANGUAGE FRONTEND

3.1 PAST WORK

Most natural language interfaces to database management systems (LUNAR, RENDEZVOUS, PLANES, LADDER, EQS, PHLIQAL, EUFID, USL, TQA, RUS/IRUS, HAM-ANS, ROBOT/INTELLECT, REL/ASK, and TED/TEAM) have approached the problem of mapping a user's query to a formal query language query by employing sophisticated parsers to recognize large, domain-independent grammars which have been coupled in some manner with domain-specific database schema information. These systems recognize a sublanguage of natural language in a limited domain. Natural language interfaces to IR systems (including SMART and CITE) have used statistical approaches with shallow parsing to translate free-form text into boolean queries in a domain-independent manner.

Only one other natural language interface system, the IRUS system developed at BBN [2], has described mixing IR and DBMS queries. IRUS allows queries like:

Find articles written by Bobrow whose subject involves syntax or pragmatics.

While the "involves" phrase appears to usher in an IR-like query, IRUS actually handles such queries by adding an extra structured field of "keywords" to an "article" entity and not by indexing textual fields of that entity.

One problem that researchers building natural language interfaces seem not to have examined is the question of how to

recognize and translate free form natural language portions of a query corresponding to IR queries when they are intermixed with limited sublanguage portions corresponding to database queries. The next section describes menu-based natural language understanding in preparation for section 3.3, which describes how IR queries were added to the system, thus solving the problem of separating the IR and DBMS portions of a query.

3.2 MENU-BASED NATURAL LANGUAGE UNDERSTANDING

In conventional query languages, the burden is on the user to understand and remember the query language syntax and semantics and also the names and relationships of the domain entities and attributes. In conventional natural language interfaces, the user types a query or command in English, and the natural language interface carries the full burden of translating the typed sentence into an operational command in the target formal language. Unfortunately, users of conventional natural language systems often have trouble phrasing their queries within the covered lexicon, syntax and semantics of the natural language interface. To overcome this limit of conventional natural language interfaces while retaining their advantage over conventional query language, we have been investigating a new approach to building natural language interfaces called menu-based natural language in an implemented system called the NLMenu system [22, 23, 25, 26].

The basic idea behind a menu-based natural language interface is that a user is presented with a constellation of menus on the upper half of a high resolution bit map display. A sample screen is shown in Figure 1. The user chooses words and phrases from the menu displays to build up a query or command. A semantically constrained grammar provides the look-ahead control structure that activates and highlights menus containing legal completions of the sentence, thus enforcing that only understandable sentences can be formed.

Conventional natural language interfaces are expensive to build and maintain in contrast to NLMenu interfaces which necessarily have smaller and more regular grammars. In the case of interfaces to relational databases, we have constructed an interface generator, previously described in [25, 26], that can automatically generate usable natural language interfaces for querying a collection of tables. The interface generator takes two inputs: 1) a domain-dependent specification called a "domain spec" which lists data dictionary information including the tables to be covered, the access rights for each table, a categorization of the attributes of tables in terms of non-numeric, numeric, and coded fields, table keys, and a specification of supported join paths, and 2) a domain-independent generic grammar and lexicon targeted on a particular database query language. The generic grammar and lexicon consist of rule templates, and the domain spec is used to instantiate the templates to generate context free grammar rules and actual lexical entries. For instance, a grammar rule

FIGURE 1: An NLMenu Interface to a University Database

NLMENU Interface Courses															
Find Delete Draw Insert	Attributes credits department title section# start-hour end-hour room instructor name spouse rank campus address extension interests department 2 course# course#2	Hours courses sections instructors interests prerequisites <specific courses> <specific sections> <specific instructors> <specific interests> <specific prerequisites> <a new course>	Experts <specific course departments> <specific titles> <specific section departments> <specific section#s> <specific start-hours> <specific end-hours> <specific rooms> <specific instructors> <specific instructor names> <specific spouses> <specific instructor ranks> <specific campus addresses> <specific extensions> <specific faculty> <specific interests> <specific prerequisite department> <specific prerequisite department number>												
	Comparisons between greater than less than greater than or equal to less than or equal to equal to	Connectors and or	Modifiers whose course department is whose course title is whose section department is whose section# is whose start-hour is whose end-hour is whose room is whose instructor is whose name is whose spouse is whose rank is whose campus address is whose extension is of which are whose prerequisite department is whose prerequisite department# is whose course course# is whose section course# is whose prerequisite course# is whose prerequisite course#2 is whose number of credits is which involve that involve which are interests of												
System Commands <div> <div>Restart</div> <div>Save Input</div> </div> <div> <div>Refresh</div> <div>Retrieve Input</div> </div> <div> <div>Rubout</div> <div>Delete Inputs</div> </div> <div> <div>Exit System</div> <div>Play Input</div> </div>															
Find courses which involve intro? and prog? and whose course department is CS															
<div>More Above</div>															
Executing Relation : NLM:COURSE Database : NLM:NLMENU Cardinality : 69.															
<table border="1"> <thead> <tr> <th>NLM:DEPAR#</th> <th>NLM:CO#</th> <th>NLM:TITLE</th> <th>NLM:CR#</th> </tr> </thead> <tbody> <tr> <td>NLM:CS</td> <td>3150.</td> <td>"INTRO TO NUMERICAL ALGOR ? PROG"</td> <td>3.</td> </tr> <tr> <td>NLM:CS</td> <td>1610.</td> <td>"INTRO TO STRUCTURED PROG"</td> <td>4.</td> </tr> </tbody> </table>				NLM:DEPAR#	NLM:CO#	NLM:TITLE	NLM:CR#	NLM:CS	3150.	"INTRO TO NUMERICAL ALGOR ? PROG"	3.	NLM:CS	1610.	"INTRO TO STRUCTURED PROG"	4.
NLM:DEPAR#	NLM:CO#	NLM:TITLE	NLM:CR#												
NLM:CS	3150.	"INTRO TO NUMERICAL ALGOR ? PROG"	3.												
NLM:CS	1610.	"INTRO TO STRUCTURED PROG"	4.												
2. tuples retrieved Execution completed.															
Nlmenu Display Window Courses															
End-of-output															

template like:

```
<rel>-mod --> whose-<rel>-<attr>-is <rel>-<attr>-expert
      where (<rel> <attr>) is an element of non-numeric-attributes
```

is instantiated with the domain spec category non-numeric-attributes:

```
non-numeric-attributes = ((course department)(course number))
```

resulting in two instantiated grammar rules, one for each of the (<rel> <attr>) pairs in non-numeric-attributes. Thus a domain-independent generic grammar is merged with a domain specification to produce a semantically constrained grammar.

The separation of the domain-dependent and domain-independent portions of an interface makes it possible for trained but linguistically naive users to build their own interfaces quickly. It took only about thirty minutes to build the NLMENU interface to the Courses Database (Figure 1). Unlike conventional portable natural language interfaces like Intellect [9], Team [8], and Ask [24], the generated interfaces are immediately usable. Conventional interfaces require a long empirical tuning phase in which a habitable application-dependent sublanguage is discovered, often requiring that a trained interface designer spend over a man-month of time to build an interface to a specific application.

3.3 EXTENDING THE NLMENU INTERFACE GENERATOR TO COVER IR QUERIES

After having defined the MATCH predicate to RTMS, only two extensions were needed to add IR query capabilities to the NLMENU interface generator. First, the domain spec defining the target application was extended by adding a new category IR-ATTRIBUTES of the form:

```
IR-ATTRIBUTES = ((entity NL_phrase attribute\s) ...)
```

For example, in the University database, the specification

```
IR-ATTRIBUTES =
  ((COURSE "whose text involves" TEXTBOOK)
   (COURSE "whose title involves" TITLE)
   (COURSE "which involve" TITLE TEXTBOOK DESCRIPTION))
```

makes it possible to restrict queries to specified fields in the generated interface as in:

```
Find courses whose textbook involves popular and
      whose title involves French and literature.
```

Second, the generic grammar was extended to include grammar rules that allowed boolean queries to be specified as follows:


```

<entity>-mod --> |<NL_phrase>-<attribute\s>| IR-query
where (<entity> <NL_phrase> <attribute\s>) in IR-ATTRIBUTES

IR-query      --> boolean-expr-popup-typein-expert
               --> boolean-expr

boolean-expr  --> boolean-expr1 (boolean-or boolean-expr)
boolean-expr1 --> boolean-expr2 (boolean-and boolean-expr1)
boolean-expr2 --> left_paren boolean-expr right-paren
boolean-expr2 --> term-expert

```

This grammar allows a user to either type a boolean search string into a popup typein window or lets NLMenu guide him through such a specification. In the NLMenu system, boolean terms can include wild cards and adjacency matching the parallel capabilities of RTMS.

Template lexical entries corresponding to the grammar terminals were also defined. Each NLMenu lexical entry is a four tuple of the form <the grammar terminal itself, the phrase to display on the NLMenu window, the window to display the phrase in, the translation of the phrase into a fragment of an RTMS call>. For instance, in the generic lexicon, the template lexical entry:

```

(<NL_phrase>-<attributes> "<NL_phrase>" MODIFIERS
 (lambda x (match x <attributes>)))
where (<entity> <NL_phrase> <attribute\s>) in IR-ATTRIBUTES

```

coupled with the value of IR-ATTRIBUTES in the above example will have the effect of placing the phrase "which involve" in the MODIFIERS window of Figure 1 and will provide the MATCH form instantiated twice in the RTMS translation of the example query above.

```

(Retreive from course
  where (and (MATCH "popular" title)
             (MATCH "French and literature" textbook)))

```

Extensions to the generic grammar and lexicon only needed to be defined once and are now available to users who wish to specify an IR-ATTRIBUTES category in the domain spec that defines their NLMenu interface. Specifying the IR-ATTRIBUTES category is all users need to do in order to gain the capability of asking mixed dbms and IR queries. Defining a KWIC index for covered IR-ATTRIBUTES, which can be done independently of the domain spec, provides an efficient access path for queries involving the covered IR fields.

It is interesting that, while, in the general case, we still cannot build the full text understanding systems of the future, two very different specific uses of natural language, one as a tightly coupled to a limited domain model (the database case) and the other loosely constrained to an associative model (the IR case) can be spliced together so cleanly using the menu-based

natural language approach. Conventional natural language interfaces would have problems in doing the same thing because of the difficulty of determining which part of the query was recognizable using the dbms-related grammar and which portion was IR-related free-text.

4. FUTURE DIRECTIONS

This paper has proposed some straightforward extensions to a relational database to support information retrieval queries, generalizing both types of systems. In addition, extensions to a natural language interface generator are detailed to allow a mix of database and IR queries with a menu-based natural language interface. More work is needed in the area of combining databases with IR systems however.

In the area of interactive query formulation, databases lag behind IR in usability. Many operational IR systems allow indices to be browsed and queries to be specified incrementally. Results of intermediate subqueries are displayed and may be cached for later use. Only a little work in the area of formatted databases has involved browsing, incremental query specification, and cooperative response [3, 7, 13, 27]. Current work on the NLMenu system involves striking a balance between efficiently optimizing a series of queries on the one hand and giving cooperative response while supporting incremental query specification on the other.

A second area where more work is needed is in hiding the details of specifying boolean queries within NLMENU natural language queries. Doszkocs [6] describes how to transform free form natural language queries into weighted boolean queries. He also describes a more complicated but practical indexing technique for weighted terms. If a (WEIGHTED_MATCH natural_language_query_string attributes) operator were implemented for use with the database, only minor changes would be needed to the NLMENU generic grammar to allow free-form natural language specifications for IR fields. This mixture of menu-driven natural language for the database portion of the query and free form natural language for the IR portion would allow a familiar natural language-like specification of queries that cross the boundary between both such systems.

ACKNOWLEDGMENTS. The authors wish to acknowledge Steve Corey and Rajini Malipatlola for their work in implementing the abstract data types facility of RTMS.

REFERENCES

- [1] Astrahan, M M, M W Blasgen, D D Chamberlin, K P Eswaran, J N Gray, P P Griffiths, W F King, R A Lorie, P R McJones, J W Mehl, G R Putzolu, I L Traiger, B W Wade, and V Watson. "System R: Relational Approach to Database Management". ACM Transactions on Database Systems, Vol 1, No 2, June 1976, pp 97-137.
- [2] Bate, Madeleine and Robert Bobrow. "A Transportable Natural Language Interface for Information Retrieval", Proceedings of the 6th Annual International ACM SIGIR Conference, Washington D C, June, 1983.
- [3] Corella, Francisco, S Jerrold Kaplan, Gio Wiederhold, and Lena Yesil. "Cooperative Responses to Boolean Queries". International Conference on Data Engineering, Los Angeles, April 24-27, 1984, pp 77-85.
- [4] Crawford, Robert and Ian Macleod. "Modular Indexing in a Relationally Based Document Retrieval System". The Canadian Journal of Information Science, vol. 6, pp 67-75.
- [5] Crawford, Robert. "The Relational Model in Information Retrieval". Journal of the American Society for Information Science, Vol. 32, No. 1, January, 1981, pp 51-64.
- [6] Doszkocs, Tamas E and B A Rapp. "Searching MEDLINE in English: A Prototype User Interface with Natural Language Query, Ranked Output, and Relevance Feedback". In: R D Tally and R R Deultgen (eds). Proceedings of the American Society for Information Science, 1979, pp 131-139.
- [7] Finkelstein, Sheldon. "Common Expression Analysis in Database Applications". 1982 SIGMOD: International Conference on Management of Data, June 2-4, Orlando, FL, pp 235-245.
- [8] Grosz, Barbara, Doug Appelt, Alex Archbold, Bob Moore, Gary Hendrix, Jerry Hobbs, Paul Martin, Jane Robinson, Daniel Sagalowicz, and Paul Martin. "TEAM: A Transportable Natural Language System", Technical Note 263, SRI International International, Menlo Park, April, 1982.
- [9] Harris, Larry. "Artificial Intelligence Corporation", In: Sondheimer, Norman (ed), Tutorial on Natural Language Interfaces, Conference on Applied Natural Language Processing, Santa Monica, 1983.
- [10] King, Richard, Henry Korth, and Barry Willner. "Design of a Document Filing and Retrieval Service", Database Week: Databases for Business and Office Applications, San Jose, May 23-26, 1983, pp 96-101.
- [11] Kolodner, Janet. "Indexing and Retrieval Strategies for Natural Language Fact Retrieval". ACM Transactions on Database Systems, Vol. 8, No. 3, September, 1983, pp 434-464.

- [12] Haskin, R L and R A Lorie, "On Extending the Functions of a Relational Database System", 1982 SIGMOD: International Conference on Management of Data, Orlando, FL, June 2-4, 1982, pp 207-212.
- [13] Kaplan, S Jerrold. "Cooperative Responses from a Portable Natural Language Query System", Phd Dissertation, University of Pennsylvania, available as HPP-79-19, Computer Science Department, Stanford, CA, July, 1979.
- [14] Macleod, Ian A. "SEQUEL as a Language for Document Retrieval". Journal of the American Society for Information Science, Vol. 30, No. 9, September, 1979, pp 243-249.
- [15] Salton, Gerard. "Suggestions for a Uniform Representation of Query and Record Content in Database and Document Retrieval". TR79-363, Department of Computer Science, Cornell University, Ithaca, New York, 1979.
- [16] Salton, Gerard and Michael McGill, Introduction to Modern Information Retrieval, McGraw Hill, 1983.
- [17] Schek, H J and P Pistor, "Data Structures for an Integrated Data Base Management and Information Retrieval System", Proceedings of the Eighth International Conference on Very Large Databases, Mexico City, September, 1982, pp 197-207.
- [18] Stonebraker, Michael, Eugene Wong, Peter Kreps, and Gerald Held. "The Design and Implementation of INGRES". ACM Transactions on Database Systems, Vol 1, No 3, September 1976, pp 189-222.
- [19] Stonebraker, M, B Rubenstein, and A Guttman. "Application of Abstract Data Types and Abstract Indices to CAD Databases". Proceedings of Database Week: Engineering Design Applications, San Jose, CA, May 23-26, 1983, pp 107-114.
- [20] Stonebraker, Michael, Heidi Stettner, Joseph Kalash, Antonin Guttman, and Nadene Lynn. "Document Processing in a Relational Data Base System". Memo UCB/ERL M82/32, Electronics Research Lab, UC Berkeley, May 6, 1982.
- [21] Salton, Gerard and Michael McGill. Introduction to Modern Information Retrieval. McGraw-Hill, 1983.
- [22] Tennant, Harry R, Kenneth M Ross, Richard M Saenz, Craig W Thompson, and James R Miller. "Menu-Based Natural Language Understanding", Proceedings of the 21st ACL, MIT, June, 1983.
- [23] Tennant, Harry R, Kenneth M Ross, and Craig W Thompson. "Usable Natural Language Interfaces Through Menu-Based Natural Language Understanding", Proceedings of the Conference on Human Factors in Computing Systems, Boston, Mass, December, 1983.

[24] Thompson, Bozena H and Fred B Thompson, "Introducing ASK, A Simple Knowledgeable System", Conference on Applied Natural Language Processing, Santa Monica, 1983.

[25] Thompson, Craig W, Harry R Tennant, Kenneth M Ross, and Richard M Saenz. "Building Usable Menu-Based Natural Language Interfaces to Databases", Proceedings of the 9th VLDB Conference, Florence, Italy, October, 1983.

[26] Thompson, Craig W. "Using Menu-Based Natural Language Understanding to Avoid Problems Associated with Traditional Natural Language Interfaces to Databases", PhD Dissertation, Department of Computer Science, The University of Texas, Austin, Texas, May, 1984.

[27] Williams, Mike, Fred Tou, Richard Fikes, A Henderson, and T Malone. "RABBIT: Cognitive Science in Interface Design". Proceedings of the Fourth Annual Conference on Cognitive Science, Pittsburgh, PA, 1982.

[28] DIALOG: Guide to Searching, Lockheed Dialog Information Retrieval Service, Palo Alto, California, November, 1979.

AUTHOR AFFILIATION: Craig Thompson is a Senior Member of Technical Staff for the Central Research Lab, Texas Instruments, Dallas, Texas.

Steve Martin has been in the MIT COOP Program at Texas Instruments, Dallas, Texas.

ADDRESS: Texas Instruments Incorporated
P.O. Box 226015, MS 238
Dallas, TX 75265

NATURAL LANGUAGE PROCESSING IN INFORMATION RETRIEVAL -- AN OUTLINE

Gerard Salton

Various methodologies have been proposed for incorporating natural language query formulations in database management and information retrieval systems. In bibliographic retrieval, the processing of natural language queries is complex, because the stored texts are not structured and the subject matter is diversified. In these circumstances, it becomes difficult to build thesauri, knowledge bases, or term association maps that are generally valid for arbitrary texts in the subject area. The use of syntactic analysis also fails to solve the query analysis problem, because syntactic methods are not adequate to make semantic distinctions.

Natural language query analysis systems are most useful in automatic relevance feedback systems where simple user inputs are incorporated in an automatic query reformulation process, and in environments where the subject matter is very tightly restricted so that useful knowledge structures spanning the subject matter can be prepared in advance.

Natural Language Processing, Query Analysis, Query Formulation,
User-System Interaction, Relevance Feedback, Knowledge-Based
Processing, Content Analysis, Information Retrieval

1. QUERY PROCESSING IN DATABASE AND TEXT RETRIEVAL

Many attempts have been made to accommodate natural language queries in database environments. [1-6] In a database application, the language analysis may be relatively tractable because:

- the queries are somewhat stereotyped since they necessarily refer to particular characteristics of the stored entities;
- the subject matter is restricted to information related to the stored entities or to the attributes characterizing the entities;
- the semantic rules can to some extent be driven by the entities and attributes included in the database.

In text based information retrieval, it becomes necessary to handle unstructured data. The attributes or characteristics of the stored texts are not easily identified and isolated, and the subject area of a given text database is not confined to a narrow topic slice. In particular,

- it is very difficult to prepare in advance a structured view of the topic area covered by a particular document collection; the basic entities which define the subject are not known in advance and neither are the relationships between entities;
- the available methods for performing automatic syntactic analyses are not sufficiently developed to produce a single correct analysis for each input text; in particular, the available methods do not solve the prepositional phrase attachment problems or the noun phrase disambiguation (for example, in the text excerpt "high frequency transistor oscillator", most syntactic analysis systems will furnish interpretations such as

"frequency transistor" and "high-frequency transistor", even though "high-frequency" actually modifies "oscillator" rather than "transistor");

- even if sufficient semantic know-how were available to reject the "high-frequency transistor" interpretation, it would still be necessary to cope with the semantic interpretation of "high" in the sense of "tall" (that is, "frequency transistor oscillators that are high (tall)", or "frequency oscillators using high (tall) transistors"). [7]

In environments where the subject area cannot be tightly circumscribed and adequate semantic know-how is not available to control the language analysis process, it is necessary to use largely nonlinguistic approaches to query processing.

2. WORD-BASED PROCESSING

In text-based retrieval, it is customary to assign to each stored item (e.g. document, paragraph, abstract, query statement, etc.) a set of content identifiers, sometimes known as keywords, terms, index terms, or descriptors. Typically certain words, or "single terms", are extracted from the available document or query texts. Additional complex terms are created from the single terms by using stored thesauruses and automatic word association techniques. Finally, weights are assigned to the terms to reflect the importance of each term for content identification purposes. A typical automatic indexing process can be carried out in the following way: [8,9]

- the word extraction procedure first eliminates from the text all common words, including all high-frequency function words ("of", "and", "or", "but", "in", etc.); the remaining words are then shortened using a suffix

deletion method to produce word stems (for example, the form "analy" could variously represent "analysis", "analyzer", "analyzing", and so on); [10]

- weights are attached to the word stems to reflect the importance of each stem for content analysis purposes based largely on word frequency consideration; one particularly useful term weighting system emphasizes terms that occur frequently inside particular document or query texts, but rarely on the outside; this scheme has come to be known as the "tf x idf" weighting system, because it consists of the product of the term frequency (tf) in a text, multiplied by the inverse document frequency (idf) in the collection; [11,12]
- complex identifications can be constructed by identifying paradigmatic relations--that is, new terms that are synonymous, or hierarchically related to the originally available terms; paradigmatic relations are often specified in a thesaurus of related words (for example, given the term "carriage", a thesaurus might lead to the terms "conveyance", or "wheel");
- additional syntagmatic relations can be picked up by studying the occurrence characteristics of pairs, or larger groups, of words in a collection, and supplying new terms associated by co-occurrence--for example, term phrases--in addition to the originally available terms (for example, "computer science" might be supplied in addition to "computer" and "science" alone.

The available experimental evidence indicates that quite simple, single-term indexing techniques can outperform much more elaborate manual, or intel-

lectual content analysis methods in terms of the amount and the quality of the retrieved materials. [13] On the other hand, the more refined automatic techniques that make use of thesauruses of related terms, and of word association maps and term phrases are not always effective. Useful thesauruses and word grouping methods are difficult to build automatically, or intellectually, and term associations that are extracted from document texts are of doubtful validity, and are in any case important only locally in the environment in which they are generated.

In the absence of useable, refined linguistic analysis techniques, it is simplest to stay with the weighted single-term indexing products, while replacing the linguistic tools by additional information obtained from the users during the course of the search operations.

3. QUERY ANALYSIS IN USER ENVIRONMENTS

One problem with current text indexing practice, both manual and automatic, is the absence of the concept of relevance of a text with respect to some query. Since the aim in text retrieval is the identification of relevant items, as opposed to extraneous ones, the notion of term relevance is important. A simple way of introducing term relevance into the retrieval process is to ask the user about the potential relevance of particular terms or phrases during the course of the search operations. One possibility for doing this consists in letting the user participate in a semi-automatic term and phrase recognition scheme: the system might first suggest certain potential terms or complex phrases based on syntactic analysis or word co-occurrence criteria; the user would then approve or disapprove any of the displayed terms before final indexing. It has been claimed that an intellectual phrase analysis system of this type could substantially improve the retrieval

effectiveness of an automatic search system. [14]

Another, potentially easy method for introducing the effect of term relevance into the retrieval process consists in using several iterative, or partial searches controlled by the well-known relevance feedback methodology. [15,16] In relevance feedback, the results of an initial search run are used automatically to reformulate the search requests by increasing the weights of query terms that are present in previously retrieved documents termed relevant to the query, and contrariwise by decreasing the weights of query terms present in the previously retrieved nonrelevant documents. In addition to changing the query term weights, the queries can also be expanded by adding new terms to the query formulations taken from the relevant documents previously retrieved. The relevance feedback process makes fewer demands on the users than a user-controlled indexing process, since the information needed from the users is confined to the submission of relevance assessments for a few previously retrieved texts.

The relevance feedback process can be applied to vector queries, where each query is formulated as a single set of query terms, and also to Boolean queries, where the query terms are related by Boolean operators. [17] It is known that relevance feedback substantially improves the retrieval effectiveness of text search systems. [15-17]

4. MODERN TEXT RETRIEVAL ENVIRONMENTS

The current text analysis and retrieval environment is characterized by imperfect linguistic analysis tools. In these circumstances, it is simplest and most effective to confine the text analysis to a single-term indexing process supplemented by a sophisticated term weighting strategy, and an iterative

search method based on relevance feedback. A more flexible text processing environment may become available in the future which could lead to enhanced text search and retrieval situations. The following possibilities must be considered in this connection:

- the use of advanced user-system interfaces and graphic display systems;
- the use of learning approaches in which system operations are improved based on experience acquired during earlier search operations;
- the use of soft, or fuzzy matching systems designed to assess the similarities between incompletely matching information queries and stored records;
- the introduction of expert system methods based on human know-how and previously stored knowledge.

None of these possibilities requires break-throughs in text understanding or linguistic analysis methodologies. As some earlier work on soft Boolean matching strategies has shown, impressive improvements are available on retrieval effectiveness even when the discourse area is incompletely specified and only rudimentary linguistic analysis methods are utilized. [18]

This study was supported in part by the National Science Foundation under grant IST 83-16166.

REFERENCES

- [1] B.W. Ballard, J.C. Lusth, and N.L. Tinkham, LDC-1: A Transportable, Knowledge-Based Natural Language Processing System for Office Environments, ACM Transactions on Office Information Systems, 2:1, January 1984, 1-25.
- [2] M. Bates and R. Bobrow, A Transportable Natural Language Interface for Information Retrieval, Sixth Annual Int. ACM SIGIR Conference, SIGIR Forum, 17:1, 1983, 81-86.
- [3] B. Grosz, TEAM: A Transportable Natural Language Interface System, Conference on Applied Natural Language Processing, Association for Computational Linguistics, Santa Monica, CA, 1983, 39-45.
- [4] L. Harris, User-Oriented Database Query with the ROBOT Natural Language System, Int. Journal of Man-Machine Studies, 9, 1972, 697-713.
- [5] G. Hendrix, E. Sacerdoti, D. Sagalowicz and J. Slocum, Developing a Natural Language Interface to Complex Data, ACM Transactions on Database Systems, 3:2, 1978, 105-147.
- [6] B. Thompson and F. Thompson, Introducing ASK - A Simple Knowledgeable System, Conf. on Applied Natural Language Processing, Association for Computational Linguistics, Santa Monica, CA, 1983, 17-24.
- [7] K. Sparck Jones and J.I. Tait, Automatic Search Term Variant Generation, Journal of Documentation, 40:1, March 1984, 50-66.
- [8] G. Salton and M.J. McGill, Introduction to Modern Information Retrieval, McGraw Hill Book Company, New York, 1983.
- [9] G. Salton, A Blueprint for Automatic Indexing, ACM SIGIR Forum, 16:2, Fall 1981, 22-38.
- [10] H.P. Luhn, A Statistical Approach to Mechanized Encoding and Searching of Literary Information, IBM Journal of Research and Development, 1:4, October 1957, 309-317.
- [11] K. Sparck Jones, A Statistical Interpretation of Term Specificity and its Application in Retrieval, Journal of Documentation, 28:1, March 1972, 11-21.
- [12] G. Salton and C.S. Yang, On the Specification of Term Values in Automatic Indexing, Journal of Documentation, 29:4, December 1973, 351-372.
- [13] G. Salton, Another Look at Automatic Text Retrieval Systems, Technical Report TR85-713, Department of Computer Science, Cornell University, Ithaca, NY, December 1985, to be published in ACM Communications, July 1986.
- [14] A.F. Smeaton, Incorporating Syntactic Information into a Document Retrieval Strategy: An Investigation, 1986 ACM Conference in Research and Development in Information Retrieval, Pisa, Italy, 1986.

- [15] J.J. Rocchio, Jr., Relevance Feedback in Information Retrieval, in The Smart System--Experiments in Automatic Document Processing, G. Salton, editor, Prentice Hall Inc., Englewood Cliffs, NJ, 1971, Chapter 14.
- [16] E. Ide, New Experiments in Relevance Feedback, in The Smart System--Experiments in Automatic Document Processing, G. Salton, editor, Prentice Hall Inc., Englewood Cliffs, NJ, 1971, Chapter 16.
- [17] G. Salton, E.A. Fox and E. Voorhees, Advanced Feedback Methods in Automatic Information Retrieval, Journal of the ASIS, 36:3, 1985, 200-210.
- [18] G. Salton, E.A. Fox and H. Wu, Extended Boolean Information Retrieval, Communications of the ACM, 26:11, November 1983, 1022-1036.

AUTHOR AFFILIATION: Gerard Salton is a Professor of Computer Science at Cornell University.

ADDRESS: Department of Computer Science
Upson Hall
Ithaca, NY 14853

**Titles of Papers Presented at the
Meeting for Which the Text Does Not
Appear in the Proceedings**

Gateway Technology by Viktor Hampel

Author Affiliation: Viktor Hampel is Program Leader of the Technology Information System (TIS) at the Lawrence Livermore National Laboratory

Address: University of California
Lawrence Livermore National Laboratory
P.O. Box 808, L-512
Livermore, CA 94550

Evaluation of Front-Ends and Gateways by Martha Williams

Author Affiliation: Martha E. Williams is a Professor of Information Science in the Coordinated Science Laboratory (CSL), Director of the Information Retrieval Research Laboratory, and an affiliate of the Computer Science Department at the University of Illinois, Urbana, Illinois.

Address: Coordinated Science Laboratory
1101 W. Springfield
Urbana, IL 61801

AUTHOR INDEX

For Papers Presented At The Conference

	<u>Page</u>
Bates, Marcia J.	285
Belkin, Nicholas J.	97
Bergman, Rita F.	235
Bernstein, Lionel M.	155
Burton, Hilary D.	295
Conry, Thomas J.	181
Corella, Francisco	73
Cotter, Gladys A.	173
Croft, Bruce	123
Davis, Dineh M.	303
Dowson, Deborah	215
Ewing, Sue M.	215
Fayen, Emily Gallup	311
Fenichel, Carol Hansen	9
Fox, Edward A.	135
Hawkins, Donald T.	275
Hushon, Judith M.	181
Jakobson, Gabriel	317
Kaplan, S. Jerrold	73
Levy, Louise R.	275
Macdonald, David B.	215
Marcus, Richard S.	213
Martin, Steve	337
Meadow, Charles T.	215
Mischo, William H.	241
Monahan, Michael	263
Pincus, Kathy W.	113
Pincus, Michael S.	113
Ramshaw, Lance A.	319

	<u>Page</u>
Salton, Gerard	355
Salveter, Sharon	329
Smith, Diane C.P.	235
Smith, Linda C.	107
Thompson, Craig	337
Thompson, Roger H.	123
Toliver, David E.	225
Walker, Edward	319
Weischedel, Ralph M.	319
Wiederhold, Gio	17
Yesil, Lana	73

SUBJECT INDEX

A listing of keywords from the Conference Papers

	<u>Page</u>
Abstract Data Types	337
Artificial Intelligence	107, 113, 123, 155, 225, 317, 329
AT&T Bell Laboratories	275
Blackboard Architecture	123, 135
BRS TERM Database	285
Common Command Language	303, 311
Content Analysis	355
Databanks	225
Database Search	215
Decision Assistance	215
Document Knowledge Representation	135
EasyNet	275
End-User Searching	241
End-Users	9, 275
Expert System	135, 155, 213
Extended Boolean Queries	135
Feedback	135
Fleet Command Center Battle Management Program	319
Front-End	9, 173, 215, 225, 235, 317, 329
Full Text Databases	285
Gastrointestinal Disease	155
Gateway Software	181
Gateways	9, 181, 215, 225, 235, 295
Information Retrieval	97, 155, 225, 337, 355
Information Retrieval Systems	235
Information Search Behavior	285
Intelligent Database Assistant	317
Intelligent Gateways	173, 295
Intelligent Interfaces	97
Intelligent Intermediary	123
Intelligent User Interface	317
Interface Design.	107
Interfaces.	17, 113, 225, 235, 319
Intermediary Information Systems	9

	<u>Page</u>
Knowledge-Based Management Systems	17
Knowledge-Based Processing	355
Knowledge System	155
KWIC Indices	337
Libraries	263
Medicine	155
Menus	135, 337
Microcomputers	181, 225
Microcomputer Interface Software	241
Multi-User	181
Natural Language Interfaces	337
Natural Language Processing	113, 135, 155, 317, 319, 329, 355
Networking	173, 263
NISO Z39G Standard Committee	311
Online	155
Online Catalogs	241
Online Interfaces	285
Online Mediation	225
Online Searching	215, 275
Online Subject Searching	285
Online Thesaurus	285
Prolog	135
Query Analysis	355
Query Formulation	355
Relational Databases	337
Relational Lexicon	135
Relevance Feedback	355
Retrieval Assistance	213
Rule-Based System	123
Search Intermediaries	9
Search Models	213
Search Modification Operators	213
Search Strategy	213
SPIRES	17
Standards	263, 303, 311
Subject Indexing	285
System Architecture	235
Technology Information System	295
User Assistance	215
User Interfaces	173, 295
User Modeling	123
User Survey	275
User-System Interaction	355

Acknowledgements

Conference Organizing Committee:

Marjorie Powell, Chair, Defense Technical Information Center
Carol Jacobson, Defense Technical Information Center
Shirley Witges, Defense Technical Information Center
Nelson Jackson, American Defense Preparedness Association
Richard Marcus, Massachusetts Institute of Technology

Compilation of Proceedings:

Carol Jacobson
Shirley Witges

Secretarial Support:

Gerry Connelly, Defense Technical Information Center